

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平6-110853

(43) 公開日 平成6年(1994)4月22日

(51) Int.Cl. <sup>5</sup>	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 15/16	3 9 0 T	9190-5L		
	4 7 0 H	9190-5L		
15/40	5 3 0 S	7218-5L		
15/80		9190-5L		

審査請求 未請求 請求項の数27(全 23 頁)

(21) 出願番号 特願平4-261108

(22) 出願日 平成4年(1992)9月30日

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72) 発明者 佐藤 裕二

東京都国分寺市東恋ヶ窪1丁目280番地

株式会社日立製作所中央研究所内

(72) 発明者 鹿山 昌宏

茨城県日立市久慈町4026番地 株式会社日

立製作所日立研究所内

(72) 発明者 前田 章

神奈川県川崎市麻生区王禅寺1099番地 株

式会社日立製作所システム開発研究所内

(74) 代理人 弁理士 小川 勝男

最終頁に続く

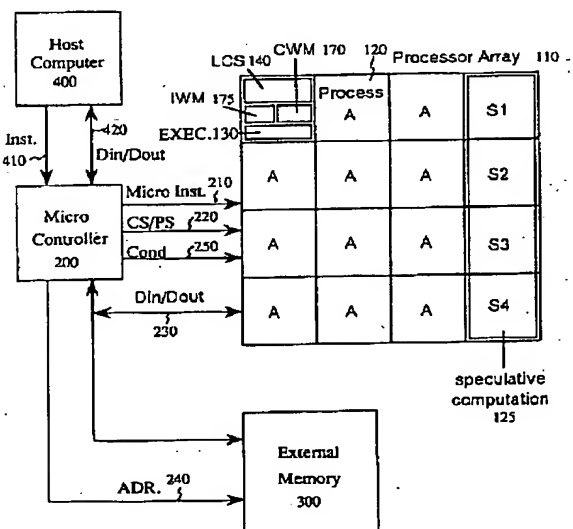
(54) 【発明の名称】 並列計算機システムおよびプロセッサ

(57) 【要約】

【構成】 複数の要素プロセッサ120から構成されるプロセッサアレイ110、プロセッサアレイ110を制御するマイクロコントローラ200を有する並列計算機システムにおいて、現在実行した結果が、将来役に立つかどうか分からない見込処理のためのプログラムを少なくとも1つを有し、複数の要素プロセッサ120のうち、少なくともいずれか1つにて見込処理のためのプログラムを実行する。

【効果】 (1) 稼働率の向上、(2) 見込処理のスケジューリング (いつ、どのような見込処理を実行させるか) の自動化、(3) 最低限必要な見込処理から、新たな見込処理の自己獲得、(4) 上記(1)~(3)に加え、例えばTATの短縮等の、付加価値の提供が実現可能である。

図 1



## 【特許請求の範囲】

【請求項1】複数のプロセッサから構成されるアレイ、および当該アレイを制御するコントローラを有する並列計算機システムにおいて、現在実行した結果が、将来役に立つかわからない見込処理のためのプログラムを少なくとも1つを有し、上記複数のプロセッサのうち、少なくともいずれか1つにて上記見込処理のためのプログラムを実行することを特徴とする並列計算機システム。

【請求項2】複数の上記見込処理のためのプログラムの1つの中、1つの見込処理のためのプログラムを特定する処理決定手段を有することを特徴とする請求項1記載の並列計算機システム。

【請求項3】上記各プロセッサまたは複数のプロセッサから構成されるグループごとに、特定の条件を判断する手段を有し、その特定の条件に従い、メインプログラムに換えて、上記特定された見込処理のためのプログラムを処理対象とすることを特徴とする請求項2記載の並列計算機システム。

【請求項4】上記特定の条件は、NOP(No Operation)命令であることを特徴とする請求項3記載の並列計算機システム。

【請求項5】上記特定された見込処理のためのプログラムを実行するための演算手段を有することを特徴とする請求項2または3記載の並列計算機システム。

【請求項6】上記演算された結果を、特定の基準に従って評価する評価手段を有することを特徴とする請求項5記載の並列計算機システム。

【請求項7】複数の上記見込処理のためのプログラムの1つの中、1つの見込処理のためのプログラムを特定する処理決定手段は、乱数表を用いて見込処理のためのプログラムを特定することを特徴とする請求項2記載の並列計算機システム。

【請求項8】上記乱数表は、擬似ランダム2進シーケンスカウンタを用いて実現することを特徴とする請求項7記載の並列計算機システム。

【請求項9】複数の上記見込処理のためのプログラムの1つの中、1つの見込処理のためのプログラムを特定する処理決定手段は、各見込処理に対応した確信度情報を有し、確信度情報に基づき実行すべき見込処理のためのプログラムを特定することを特徴とする請求項2記載の並列計算機システム。

【請求項10】上記各見込処理に対応した確信度情報を、上記評価手段に従い、オンライン（稼働）中に更新する手段を有することを特徴とする請求項9記載の並列計算機システム。

【請求項11】上記演算された結果を記憶する手段を有し、当該見込処理結果を検索し、特定の基準に従う結果を選択する手段を有することを特徴とする請求項5記載の並列計算機システム。

【請求項12】上記見込処理結果を記憶する手段へ格納する情報は、検索を容易にするためのラベリング情報等を演算結果に付加して格納することを特徴とする請求項11記載の並列計算機システム。

【請求項13】上記特定されたプログラムを実行するための演算手段は、NOP(No Operation)状態の要素プロセッサを活用し実行することを特徴とする請求項5記載の並列計算機システム。

【請求項14】上記見込処理として、予想される幾つかの相手の回答に対して、次の自分の回答を予め準備するという処理を想定し、対話形式で処理が進行し、かつ、選択できる回答が予め限られている囲碁・将棋等のゲームや対話型コンピュータに適用することを特徴とする請求項1記載の並列計算機システム。

【請求項15】上記見込処理として、NOP部のオンライン自己テストを想定し、ニューロコンピュータ、及び超並列計算機の信頼性向上に適用することを特徴とする請求項1記載の並列計算機システム。

【請求項16】上記テストの結果故障が検出された場合、故障ユニットであるボード、チップ、またはプロセッサを並列計算機システムから自動的に分離（自己修復）する機能を有することを特徴とする請求項15記載の並列計算機システム。

【請求項17】上記故障ユニットを並列計算機システムから自動的に分離（自己修復）する手段は、プロセッサ間の結合強度を記憶するメモリの値を書き替えることにより実現することを特徴とする請求項16記載の並列計算機システム。

【請求項18】上記見込処理として、検索処理における最終的な処理（または検索結果）を想定し、検索処理のTAT短縮に適用することを特徴とする請求項1記載の並列計算機システム。

【請求項19】上記見込処理として、危険を伴う処理において将来起こる可能性のある処理を想定し、危険予知または危険回避に適用することを特徴とする請求項1記載の並列計算機システム。

【請求項20】上記見込処理として、アミノ酸またはアミノ酸を表現するデータを想定し、目的に合ったタンパク質合成に適用することを特徴とする請求項1記載の並列計算機システム。

【請求項21】上記見込処理として、遺伝子または遺伝子を表現するデータを想定し、目的に合った遺伝子を絞り込むことに適用することを特徴とする請求項1記載の並列計算機システム。

【請求項22】上記見込処理のためのプログラムを基に、新たなプログラムを自律的に獲得する手段を有することを特徴とする請求項1記載の並列計算機システム。

【請求項23】上記見込処理のためのプログラムを基に新たなプログラムを自律的に獲得する手段は、上記見込処理のためのプログラムを複数回繰り返した結果、特定

の基準を満たす結果を得るまでの上記見込処理のためのプログラムの組合せを新たなプログラムとして記憶装置に格納することにより実現することを特徴とする請求項22記載の並列計算機システム。

【請求項24】所定の制御または初期値決定を目的とするプロセッサにおいて、必ずしも有効性が保証されない少なくとも1つのデータを処理対象とすることを特徴とするプロセッサ。

【請求項25】上記必ずしも有効性が保証されない少なくとも1つのデータを記憶する手段を有することを特徴とする請求項24記載のプロセッサ。

【請求項26】上記必ずしも有効性が保証されない少なくとも1つのデータは、同一環境条件における、前回ユーザ指定値であることを特徴とする請求項24記載のプロセッサ。

【請求項27】上記環境条件として、室温、湿度、時刻、曜日の中の少なくとも一つは含んでいることを特徴とする請求項26記載のプロセッサ。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、一般的には複数のプロセッサから構成される超並列計算機などの並列計算機システムに関するものである。より詳細には、複雑な情報処理に対応するための超並列計算機の使用法および要素プロセッサの構成に関するものである。

【0002】本発明は、ニューロコンピュータにも適用可能である。また、マイクロプロセッサ単体としても利用可能である。

【0003】

【従来の技術】伝統的なコンピュータシステムは、プログラムメモリから単一の命令ストリームを入力し、1度に1つの命令を実行する単体のプロセッサから構成されている。これは、順次単一プロセッサ・アーキテクチャ（フォンノイマン型アーキテクチャ）として知られている。

【0004】順次単一プロセッサ・アーキテクチャにおける高速化の一つの手法として、例えば、68000マイクロコンピュータ（丸善、1983年3月発行、P. 19-20）に命令プリフェッチ（命令の先取り）機能が記載されている。この機能を実現するための簡単なレジスタ構成を図18に示す。

【0005】図18において、現在実行中の命令コードはIR（Instruction Register）820にセットされ、命令がデコードされた後、信号線821を介してIRD（Instruction Decoder）810に移される。命令がIRD810に移されると次に実行される命令がIRC（Instruction Cache）830から、信号線831を介して、IR820に移されデコードの開始を待つ。原則としてIR820、IRC830には次に実行される

命令の第一ワード目が各々セットされる。命令の第2ワード目からの拡張部分はEDB（Entry Data Bus）840にセットされる。このように、プリフェッチ用の特別なレジスタを有することにより、次に実行される命令の全体または一部を先取り（メモリからの読み出し）し、実行的に命令フェッチ時間を節約することが可能である。

【0006】しかし、上記伝統的なコンピュータシステムにおける単一プロセッサの性能向上だけでは、ユーザが望む計算速度や処理能力が得られなくなっている。この対策として、多数の要素プロセッサを並列実行させることにより高速化を実現する並列計算機（超並列計算機）システムが注目されてきている。一例として、特開昭60-84661号公報に記載されている、シンキング・マシンズ社のコネクションマシンのシステム構成概略を図17に示す。

【0007】図17において、110はプロセッサアレイ、120はプロセッサアレイを構成するプロセッサである。200はプロセッサアレイを制御するマイクロコントローラ、300は演算に必要なデータを格納する外部記憶装置、400はマイクロコントローラ200を制御するホストコンピュータである。

【0008】ホストコンピュータ400は、例えばDEC（Digital Equipment Corp.）社製造のVAXコンピュータのような市販されている汎用コンピュータであり、制御信号410によりマイクロコントローラ200の動作開始・終了を指示する。また、マイクロコントローラ200を介して、信号線420により外部記憶装置300またはプロセッサアレイ110との間でデータの送受信を行う。

【0009】マイクロコントローラ200は、並列バス210により、プロセッサアレイ110の制御を行う。例えば、バス210中の1回線はアレイ110にRESET信号を与え、3回線はタイミング信号を出し、残りの回線は命令の伝達に使われる。対象となるチップおよびプロセッサは信号線220で行う。外部記憶装置への読み書きの為のアドレス情報はアドレス信号240により行う。マイクロコントローラ200とプロセッサアレイ110との間のデータの送受信およびプロセッサアレイ110からマイクロコントローラ200への状態情報の通知は、信号線230により行う。

【0010】バス210および230は、各要素プロセッサ120に並列に接続される。その結果、マイクロコントローラ200からの信号は、アレイ110内の各要素プロセッサ120に同時に与えられる。すなわち、各要素プロセッサ120は一度に同一命令を実行する。全プロセッサを必要としない場合は、余ったプロセッサ125はNOP（No Operation）状態となる。

【0011】NOP状態のプロセッサ数の増加はコストパフォーマンスの低下につながる。従って、NOP状態

のプロセッサをできるだけ少なくするタスク（負荷）割当て技術、例えば、一定期間以上NOPが続いたら負荷を割当てる技術が、特開平3-191488号公報等に記載されている。

#### 【0012】

【発明が解決しようとする課題】一般的に、より大規模な問題あるいはより複雑な情報処理に対応するためには、超並列計算機システムを構成する要素プロセッサの数を増加させる必要がある。

【0013】並列度を上げて高速化を実現する場合も、10 プロセッサ数を増加させる必要がある。しかし、応用プログラムによっては、必ずしもシステム内の全プロセッサを必要としない。また、メインプログラムを構成するサブプログラム（タスク）実行時には、メインプログラムが必要とする要素プロセッサの一部しか使用しない。すなわち、上記従来技術ではシステムを構成するプロセッサ数の増加に伴い、稼働率が低下する（NOP状態のプロセッサの割合が増加する）という問題点が存在する。この傾向は、一度に全プロセッサが同一命令を実行するSIMD(Single Instruction stream Multi Data stream)型アーキテクチャの並列計算機では特に顕著である。

【0014】また、より複雑な情報処理に対応するためには例えば人間の持つ先読み機能（先の動きを予測する機能あるいは将来に備えて予め見込処理（役に立たないかも知れないが、もしかしたら後で役に立つかも知れない処理）を実行する機能）が有効と思われるが、上記従来技術（例えば命令プリフェッチ機能）は既に確定した処理アルゴリズムにおいて次に実行すべき命令を予めメモリから読み出しておくだけ（処理の実行はしない）であり、人間の持つ先読み機能には十分対応出来ない。上記従来技術において先読み機能を実現するためには、いつ、どのような先読み機能を実行させるかを人が考えてメインプログラムを書く必要がある。すなわち、先読み機能のスケジューリングを含めたアルゴリズムの検討が必要であり負担が大きい。また、応用プログラムごとに上記スケジューリングを再検討する必要があり、プログラム作成工数が大幅に増加し、バグ発生確率も大きくなるという問題点が存在する。

【0015】上記従来技術の問題点を解決するために、40 本発明では以下に示す(1)～(4)を目的とする。

【0016】(1) 超並列計算機（特にSIMD型アーキテクチャ）の稼働率の向上

(2) 上記先読み機能のスケジューリング（いつ、どのような先読み機能を実行させるか）の自動化（自己形成）

(3) 最低限必要な見込処理から、新たな見込処理の自己獲得

(4) 上記(1)～(3)に加え、例えばTATの短縮等の、付加価値の提供

#### 【0017】

【課題を解決するための手段】上記目的を達成するために、以下に示す解決手段を有する。

【0018】(1) 実行結果が（後で）役に立つかどうか分からない複数のプログラム（見込処理）を、メインプログラムとは別に、記憶する手段を有する。

【0019】(2) 上記複数の見込処理対象の中から、1つのプログラムを特定する手段を有する。例えば、上記見込処理に対して、各外部条件（実行条件）ごとに、各見込処理の確からしさをしめす情報（重み値）を有し、重み値情報に従い1つのプログラムを特定する。あるいは、乱数表を用いて1つのプログラムを特定する。

【0020】(3) 上記特定された見込処理プログラムを実行するための演算手段を有する。

【0021】(4) 上記見込処理の実行結果を、特定の基準に従って評価する手段を有する。

【0022】(5) 各要素プロセッサまたは複数の要素プロセッサから構成されるグループごとにNOP(No Operation)状態を判断する手段を有する。

【0023】(6) 上記NOP状態判定結果に従って、メインプログラムに換えて、上記特定された見込処理プログラムを実行する手段を有する。

【0024】(7) 上記見込処理の評価結果に従い、上記重み値情報をオンライン（稼働）中に更新する手段を有する。

【0025】(8) 上記（同一または異なる）見込処理を複数回繰り返す手段を有する。

【0026】(9) 上記繰り返した見込処理の来歴を記憶する手段を有する。

【0027】(10) 複数の見込処理の組み合わせを、新たな見込処理として、記憶する手段を有する。

#### 【0028】

【作用】上記問題解決手段により、下記に示す作用を得る。

【0029】(1) 本発明によれば、メインプログラムとは別に、複数の見込処理プログラムをシステム内に記憶し、該複数の見込処理プログラムの中から1つを特定した上で、各要素プロセッサにおいて、NOP状態時には、メインプログラムに換えて、上記特定された見込処理プログラムを演算可能である。

【0030】従って、NOP状態のプロセッサを見込処理の実行に活用することにより、稼働率が向上する。適当な見込処理が実行されれば、例えば検索処理等において、TATの短縮につながる。また、仮に無効な見込処理が実行されたとしても、もともとNOP状態のプロセッサを用いているために、性能低下にはならない。

【0031】(2) 本発明によれば、見込処理の実行結果を特定の基準に従って評価し、該評価結果に従い、各外部条件（実行条件）ごとに、各見込処理の確からしさをしめす情報（重み値）をオンライン中に更新する。

50 【0032】従って、使えば使うほど（経験を通し

て)、外部条件(実行条件)に対応して、的確な見込処理が選ばれるように自己成長可能である。

【0033】(3) 本発明によれば、見込処理を複数回繰り返した結果有効な解(収束解)が得られた場合、有効な解(収束解)を得るまでに実行した一連の見込処理の組み合わせを新たな見込処理として利用可能である。

【0034】従って、予め人間が与えた見込処理(遺伝的要素)を組み合わせ、新たな見込処理(ある初期状態、ある入力データに対して有効な解(収束解)を与える一連の見込処理の流れ(組み合わせ))を自己獲得可能である。

【0035】

【実施例】以下、本発明の実施例を図1から図20により説明する。

【0036】図1は本発明の一実施例を示すためのSIMD (Single Instruction stream Multi Data stream) 型アーキテクチャの超並列計算機のシステム構成図である。

【0037】図1において、110はプロセッサアレイ、120はプロセッサアレイを構成するプロセッサである。200はプロセッサアレイを制御するマイクロコントローラ、300は演算に必要なデータを格納する外部記憶装置、400はマイクロコントローラ200を制御するホストコンピュータである。

【0038】ホストコンピュータ400は、例えば制御信号410によりマイクロコントローラ200の動作開始・終了を指示する。また、マイクロコントローラ200を介して、信号線420により外部記憶装置300またはプロセッサアレイ110との間でデータの送受信を行う。

【0039】マイクロコントローラ200は、マイクロ命令210により、プロセッサアレイ110の制御(例えば、演算機能の指定)を行う。対象となるチップ(プロセッサアレイまたはプロセッサアレイの一部)およびプロセッサはチップセレクト信号CSおよびプロセッサセレクト信号PS220で行う(図1では、プロセッサセレクト信号PSの一部分(例えば、上位数ビット)をチップセレクト信号CSとして用いることを仮定しているために、CSとNSを同一信号220で表している。別信号として実現することも可能である。)。外部記憶装置への読み/書きのためのアドレス情報はアドレス信号240により行う。マイクロコントローラ200とプロセッサアレイ110との間のデータの送受信は、Din/Dout信号230により行う。

【0040】図1において図17に示す従来のSIMD型超並列計算機との違いは、プロセッサ120を構成する要素として、演算器群130の他に、局所記憶装置140、局所記憶装置140内に格納した複数の情報(例えば、プログラム)の確信度を示すインストラクション・ウェイト・メモリ(IWM)175、プロセッサ間の

結合強度を示すコネクション・ウェイト・メモリ(CWM)170を有する点にある。Cond250は外部条件(または実行条件)であり、IWM175のポインタとなる。

【0041】以下、図2および図3により本発明の特徴を示す。

【0042】図2は、図1に示した要素プロセッサ120のブロック構成図の一例である。

【0043】図2において、130、140、175はそれぞれ図1に示した演算器群、局所記憶装置、IWMである。図2では、局所記憶装置140には実行結果が必ずしも(後で)役に立つかわからない複数のプログラム(見込処理)を記憶する。IWM175には、各外部条件(実行条件)ごとに、各見込処理の確からしさを示す情報(重み値)を格納する。166はNOP (No Operation) 判定論理、165は(NOP判定論理、局所記憶装置およびIWMからの情報をもとに)演算器群130の制御信号を生成する論理、190は実行した命令の有効性を判定する論理である。

【0044】以下、図3に示したフローチャートに基づき図2に示すプロセッサの特徴的な動きを示す。

【0045】NOP判定論理は、マイクロコントローラ200から入力したCS/PS220、マイクロ命令210および演算器群130から入力したフラグ情報162から、NOP判定を行う。また、判定結果163をINST、決定論理165に出力する。

【0046】INST、決定論理165は、判定結果163がNOP指示でなければマイクロコントローラ200から入力したマイクロ命令210を演算器群制御用の命令161として出力する。(すなわち、演算器群130ではメインプロセスを実行する。)判定結果163がNOP指示の場合は局所記憶装置140に格納した複数の(図2では、141-1~141-4の4つ)の見込処理の中から1つを選択し演算器群制御用の命令161として出力する。また、選択した命令の情報を選択の成否判定論理190に出力する。図2では、IWMから読み出した、各見込処理の確からしさを示す重み値情報(176~179)からどの見込処理を選択するかを決定する。重み値の最大値は、例えば、信号線169を介して各重み値情報を演算器群130に入力して求める。IWMの初期値は例えば乱数表により(ランダム)に与える。あるいは、意図を持ち、人が決定する。IWMのポインタはマイクロコントローラ200から入力した外部条件(実行条件)250を用いる。

【0047】演算器群130は、INST、決定論理165から入力した命令161に従い演算を実行し、結果(および選択した見込処理情報)を、例えば、レジスタ等の記憶装置に格納する。演算に必要なデータは、例えばDin230を介して予めマイクロコントローラ200または外部メモリ300から入力し、演算器群130内

の記憶装置に格納しておく。

【0048】選択の成否判定論理190では、例えば、本来実行すべきだった命令の情報230と選択した命令164から演算結果131の有効性を判定する。有効な場合は演算結果を、信号線192を介して他のプロセッサなどに送信し、メインプロセスで利用する。また、選択した命令の成否判定結果191を演算器群130に出力する。

【0049】演算器群130は、成否判定結果191に従い、選択した命令に対応した重み値情報の更新を行\*10【表1】。

表1. NOP判定論理

入力			出力
CS/PS信号220	マイクロ命令番号210	フラグ信号162	NOP判定結果信号163
0	*	*	1
1	NOP	*	1
	NOP	0	0
	以外	1	1

\*：ドントケア

【0053】CS/PS信号は、信号値が'1'のとき選択、'0'のとき非選択。フラグ信号は、'1'のときNOP指示。NOP判定結果信号は、'0'のときマイクロコントローラ200からの命令210を選択指示、'1'のとき見込処理を選択指示することを意味する。

【0054】図4に演算器群130のブロック構成例を示す。図4において、121は入力バッファ、122は出力バッファであり、それぞれ他の要素プロセッサから入力したデータ、他の要素プロセッサへ出力するデータを一時的に格納する機能を有する。131はシフト機能付き算術論理演算ユニット、132は乗算器、133は汎用レジスタ、134はフラグレジスタ、136-1~136-2はパラメータレジスタである。また、170はプロセッサ間の結合強度を示す重み値情報を格納するための書き換え可能なメモリである。136-1~136-2および170は汎用レジスタ133の一部に割当てられることも可能である。137~139は内部バスである。230-1、169、191はそれぞれ図2に示すDin/Dout、重み値情報、選択の成否結果である。230-2、131、132、162はそれぞれ図2に示すDin/Dout、演算結果、更新後の重み値情報、フラグ情報である。

【0055】図5にINST. 決定論理165の一例を示す。INST. 決定論理165は、NOP判定結果信号163が'0'のときマイクロコントローラ200から入力したマイクロ命令210を演算器群130制御用の命令161として出力する。

\*う。例えば、正しい選択であった場合は重み値を増加させる。誤った選択であった場合は重み値を減少させる。更新後の重み値は信号線132を介してIWM175に格納する。

【0050】次に、表1および図4~図7を用いて、図2に示す要素プロセッサの構成要素の詳細を示す。

【0051】表1はNOP判定論理の詳細を示す真理値表である。

【0052】

【表1】

【0056】NOP判定結果信号163が'1'のときは局所記憶装置140から見込処理プログラム141-1~141-4の中から1つを選択し演算器群130制御用の命令161として出力する。また、選択した命令の情報164を選択の成否判定論理190に出力する。図5では演算器群130制御用の命令161をそのまま選択した命令の情報164として流用している。図5では、IWM175から読み出した、各見込処理の確からしさを示す重み値情報176~179からどの見込処理を選択するかを決定する。重み値の最大値は、信号線169を介して各重み値情報を演算器群130に入力して求める。例えば図4に示した演算器群の中のALUを用いて2つの重み値の差を求め、結果が正か負かで大小を判定する。重み値情報176~179の中の最大値が得られたら、結果を最大値レジスタ168に登録する。図5では、最大値に対応したレジスタに'1'をセットしている。

【0057】図6に、選択の成否判定論理190の一例を示す。図6において195はコンパレータであり、本来実行すべきだった命令の情報230と選択した命令164を比較し、一致した場合信号線191に値'1'を出力。不一致の場合'0'を出力する。信号線191の値は、成否結果として演算器群130に出力される。また、演算結果131を信号線192を介して他の要素プロセッサに伝達するかを制御する信号として用いられる。コンパレータ195の替わりにALUを用いることも可能である。選択の成否判定論理190（コンパレータ195）は、演算器群130を用いても実現可能であ

る。

【0058】図7に重み値(IWM)更新処理の一例を示す。図7では、演算器群130の一部である乗算器132とパラメータレジスタ136-1~136-2を用いて重み値を更新する例を示している。レジスタ136-1と136-2には予め、 $0 < a < 1$ ,  $1 < b$ を満たす、値a, bをそれぞれ格納する。選択の成否判定論理190から入力した成否結果191の値が'0'のときは、パラメータレジスタ136-1に格納されている値aと選択した見込処理に対応した重み値Wiをそれぞれ内部バス138と137を介して乗算器132に入力し、演算結果をバス139を介してIWM175に格納する。成否結果191の値が'1'のときは、レジスタ136-2に格納されている値bと重み値Wiをそれぞれ内部バス138と137を介して乗算器132に入力し、演算結果をバス139を介してIWM175に格納する。

【0059】次に、図8から図13により第2の実施例を示す。

【0060】図8はメインプログラムとはほとんど独立した処理を実行するためのブロック構成図の一例である。

【0061】図8において、130は演算器群、140は見込処理を格納するための局所記憶装置、165はInst. 決定論理、1133は演算結果を一時的に格納するための書き換え可能な記憶装置である。

【0062】Inst. 決定論理165は、NOP判定結果163がNOP指示の場合、マイクロコントローラ200から入力したマイクロ命令210に換えて、局所記憶装置140から入力した見込処理を実行する。図2および図3で示した処理との違いは、メインプログラムとはほとんど独立した見込処理を想定しているために、各見込処理の確からしさを示す重み値情報(IWM)175を持たず、実行すべき見込処理を、例えば乱数表あるいは擬似ランダム2進シーケンスカウンタ等を用いて、ランダムに選択している点にある。

【0063】演算器群130により処理された見込処理の結果は、visionary register 1133に格納され、適当なタイミングで検索される。検索の結果たまたま有効な情報が存在すれば、メインプロセスで活用される。visionary register 1133が一杯になったら、古い情報から削除(忘却)する。あるいは、アクセス頻度の少ない情報から削除(忘却)する。visionary register 1133は、例えば選択した見込処理情報等を検索キーとして、ラベル付けしておくことと高速な検索に有効である。

【0064】図9は、上記見込処理を複数回繰り返すことにより、新たな見込処理を自律的に獲得するモデルを示している。以下、図10に示したフローチャートに基づき図9に示す自律的な見込処理獲得を示す。

【0065】図9において、910はメインプロセスで

あり、通常、入力911をもとに定型的な処理(アルゴリズムが確定しているプログラム)を行い、結果912を出力する。

【0066】NOPの場合は、局所記憶装置140内の複数の見込処理の中から1つをランダムに選択し、入力データ911を必要に応じて利用しながら、選択した見込処理951を実行する(930)。実行結果および選択した見込処理情報931はVisionary Memory 970に格納される。

【0067】Visionary Memory 970に格納した実行結果が定常解または有効な解でない場合はこの解を新たな入力データ971として、局所記憶装置140から新たな見込処理を1つランダムに選択し実行する。あるいは、先に選択した見込処理を再度実行する。以上の処理を一定期間繰り返す。一定期間繰り返す中でたまたま定常解または有効な解が得られたら、その解をメインプロセスで利用する。また、定常解(または有効な解)を得るまでの見込処理のシーケンス972を新たな見込処理として局所記憶装置140に登録する。

【0068】図11に自律的な見込処理獲得の例を示す。図11では、見込処理の初期値として文字を与え、メインプロセスでえた知識を評価関数としながしだいに文字から単語、単語から文章を新たな見込処理として獲得していく流れを示している。

【0069】同様な応用例として、タンパク質合成も考えられる。見込処理としてアミノ酸を考え、見込処理を繰り返す(アミノ酸を組み合わせる)という処理を行い、目的(外部から与える評価関数)に合ったタンパク質を形成するアミノ酸の組み合わせを得ることができる。一種の自己組織化と考えられる。

【0070】図9はまた、見込処理の絞り込みモデルと考えることもできる。例えば、見込処理として遺伝子を考える。見込処理を繰り返す(遺伝子を組み替える/組み合わせる)という処理を行い、目的(例えば、病害虫に強い、高栄養、安全等の評価基準)に合う見込処理(遺伝子、または遺伝子の組み合わせ)のみを残す(絞り込む)ことができる。一種のGenetic Algorithm(遺伝的アルゴリズム)と考えられる。

【0071】次に、図12および表2によりVisionary Memoryに格納する情報の一例を示す。図12は、Visionary Processの遷移の例を示している。図12では、見込処理としてs1およびs2の2つのみを考え、初期状態Sに入力i1を与えた後、s1またはs2を3回繰り返した後、r1~r4の収束解が得られたことを示している。

【0072】表2は、上記図12に対応して、Visionary Memoryに格納する情報の一例を示している。

【0073】

【表2】



表2. Visionary Memoryに格納する情報の一例

初期状態	入力	遷移	結果
S	i 1	0 0 0	r 1
S	i 1	0 1 0	r 2
S	i 1	1 0 0	r 3
S	i 1	1 1 1	r 4
:	:	:	:
:	:	:	:
:	:	:	:

遷移0: 見込処理s 1, 遷移1: 見込処理s 2

【0074】例えば、表2の一行目は初期状態Sに入力i 1が与えられた後、遷移000（すなわち、見込処理s 1, s 1, s 1）が実行され、結果として収束解r 1が得られたことを示している。

【0075】適当なタイミングでVisionary Memoryを検索し、収束解に対応した遷移（見込処理のシーケンス）を局所記憶装置140に登録することにより新たな見込処理が獲得されたことになる。

【0076】図13は、図9に示す自律的な見込処理獲得モデルの変形例である。図13では、図2で示したある程度メインプロセスと因果関係を考えた見込処理と、図8で示したメインプロセスとの関係をほとんど考えない見込処理が共存する場合を想定し、前者を見込処理、後者を空想的処理と呼んでいる。

【0077】図13において、910はメインプロセスであり、通常、入力911をもとに定型的な処理（アルゴリズムが確定しているプログラム）を行い、結果912を出力する。

【0078】NOPの場合は、局所記憶装置140-1内の複数の見込処理の中から1つをランダムに選択し、入力データ911を必要に応じて利用しながら、選択した見込処理951を実行する（920）。実行結果および選択した見込処理情報921はSpeculative Memory960に格納される。

【0079】Speculative Memory960に格納した実行結果が定常解または有効な解でない場合はこの解を新たな入力データ961として、局所記憶装置140-1から新たな見込処理を1つランダムに選択し実行する。あるいは、先に選択した見込処理を再度実行する。以上の処理を一定期間繰り返す。一定期間繰り返す中でたまたま定常解または有効な解が得られたら、その解をメインプロセス910で利用する。また、定常解（または有効な解）を得るまでの見込処理のシーケンス962を新たな見込処理として局所記憶装置140-1に登録する。

【0080】あるいは、NOPの場合は、局所記憶装置140-2内の複数の空想的処理の中から1つをランダムに選択し、入力データ911を必要に応じて利用しながら、選択した空想的処理952を実行する（930）。実行結果および選択した空想的処理情報931はVisionary Memory970に格納される。

【0081】Visionary Memory970に格納した実行結果が定常解または有効な解でない場合はこの解を新たな入力データ971として、局所記憶装置140-2から新たな空想的処理を1つランダムに選択し実行する。あるいは、先に選択した空想的処理を再度実行する。以上の処理を一定期間繰り返す。一定期間繰り返す中でたまたま定常解または有効な解が得られたら、その解をメインプロセス910で利用する。また、定常解（または有効な解）を得るまでの見込処理のシーケンス972を新たな見込処理として局所記憶装置140-1および140-2に登録する。

【0082】以上、図1から図13に示した実施例では、見込処理（空想的処理）の実行にNOP状態のプロセッサを活用する例を示した。これは、仮に無効な見込処理（空想的処理）が実行されたとしても、もともとNOP状態のプロセッサを用いているために、性能低下にならないからである。

【0083】NOPの代わりに、何らかの外部条件（実行条件）を認識し、見込処理（空想的処理）を実行することも可能である。見込処理（空想的処理）用に、専用のプロセッサを割り当てることも可能である。

【0084】以上、図1から図13に示した実施例では、SIMD型計算機を例にあげたが、特にSIMD型計算機に限定するものではない。MIMD型計算機でも同様の効果が期待できる。

【0085】また、見込処理を格納する局所記憶装置140は、特にプロセッサ単位に有する必要はない。プロセッサグループ（例えば、チップやボード）単位に有す



ることも可能である。

【0086】上記本発明は、応用分野（メインプロセス）および想定する見込処理（空想的処理）により、種々な適用が考えられる。以下、図14から図16により本発明の適用例を示す。

【0087】図14は、人とマシンあるいはマシン対マシンの間で、対話形式で進行する処理への適用例である。図14において、1400は人（またはマシン）、1500はマシンである。1401～1421は人1400からマシン1500への応答、1501～1531はマシン1500から人1400への応答を示している。1410、1420および1510～1530は入力に対して判断を下し、何らかの処理を行うことを示している。例えば、1420では、入力 $B_{i-1}$  1522に対して下した判断に対応して関数 $f_i$ を選択し、関数変換 $f_i$ を入力 $B_{i-1}$ に施した結果 $A_i$  1421をマシン1500に返す。1530では、入力 $A_i$  1421に対して下した判断に対応して関数 $g_i$ を選択し、関数変換 $g_i$ を入力 $A_i$ に施した結果 $B_i$  1531を人1400に返す。

【0088】ここで、選択できる関数が予め限られている世界、例えば囲碁や将棋等、では次に出力する自分の応答に対して相手の回答（応答）が限定される。従って、“予想されるいくつかの相手の回答に対して、次の自分の回答を、NOP状態のプロセッサを活用して、予め準備”という見込処理を（1回以上）繰り返すことにより、うまく予想が当れば、TATの短縮となる。あるいは、このような見込処理を繰り返した結果、“（自分にとって）最良の結果を導くためには”という観点から、最初の自分の回答を決定することにより複雑な機能の実現あるいは思考を中断させない対話の流れを実現可能となる。

【0089】図15はニューロコンピュータの自己テストへの適用例である。図15に示すニューロチップ110は、複数（図15では2個）の要素PE120、局所記憶装置140、テストデータ記憶装置150およびチップ内故障フラグレジスタFRC180により構成されている。図15では、見込処理としてオンライン（稼働）中の自己テストのみを想定し、テスト用のマイクロ命令を通常動作用のマイクロ命令210（マイクロコントローラから入力）とは独立に各ニューロチップ110内の局所記憶装置140に分散配置し、かつ、通常動作用のマイクロ命令210とテスト用の命令141を選択実行可能な点にある。局所記憶装置140およびテストデータ記憶装置150は、いずれも書き換え可能なメモリ（例えば、RAM、EPROM、レジスタ等）で構成され、アプリケーションあるいはタスク（処理）により書き換え可能である。図15において、要素PE120は、演算器群130、入力バッファ121、出力バッファ122、制御信号決定論理160およびニューロン間

の結合強度である重み値を格納する重み値記憶装置CWM170により構成されている。制御信号決定論理160は、図2に示すInst. 決定論理165とNOP判定論理166を合わせた論理である。演算器群130は、シフト機能付き算術論理演算ユニット131、乗算器132、レジスタ群133、フラグレジスタ134、PE内故障フラグレジスタFR135により構成されている。図15では、見込処理としてオンライン（稼働）中の自己テストのみを想定しているために、各見込処理の確からしさを示す重み値情報を格納するメモリIWM175は有しない。

【0090】図15を用いて動作概略を説明する。例えば、CS220が1（通常動作）のとき、制御信号決定論理160はマイクロコントローラ220から入力したマイクロ命令210を演算器群130を制御するための命令161として出力する。演算器群130は、制御信号161に従い、例えばDin230-1から入力バッファ121を介して入力される他要素PE（ニューロン）からの入力データとCWM170から信号線171を介して入力される重み値データからニューロンの内部状態を計算する。あるいは、重み値の更新処理を行う。他PE（ニューロン）へのデータの出力は、出力バッファ122を介してDout230-2を用いて行う。一方、CS220が0（当該チップは非選択）のとき（あるいはCS220が1かつフラグレジスタからの入力162が特定の値（例えば1のとき）、制御信号決定論理160は局所記憶装置140から信号線141を介して入力したテスト用のマイクロ命令を演算器群130を制御するための命令161として出力する。演算器群130は、制御信号161に従い、例えばテストデータ記憶装置150から信号線151を介してテストデータを入力し、各演算器群の機能テストを行う。演算結果の期待値は予め、例えばREG133の一部（アドレス固定）に格納しておく。テストの結果故障が検出されない場合、FR135は初期値（例えば0）を保持する。故障が検出された場合、FR135は値を更新（例えば1にセット）する。また、重み値記憶装置170内の重み値の値を0に設定する。すなわち、故障発生PE（ニューロン）をシステムから分離する。各PE内故障フラグレジスタFR135の値はOR論理をとり、チップ内故障フラグレジスタFRC180にセットされる。FRC180の値は故障検出信号181としてマイクロコントローラ200に伝達される。マイクロコントローラ200は、故障検出信号181により故障発生を検知すると、ホストコンピュータ400に対して、故障対策指示を与える。ホストコンピュータ400は、故障対策指示に従い、例えばユーザに対して故障発生通知および故障発生個所の表示を行う。あるいは、故障ユニット（ボード、チップまたはPE）情報をコンパイラに与え、故障ユニットにタスク（処理）が割当てられないようにする。

【0091】以上図15により、オンライン状態でのNOP部の自己テストへの適用例を示した。テストした部分が十分短い時間の後稼働状態となれば、オンラインテストとして有効である。稼働状態となるまでの時間間隔が長ければ、オンラインテストとしての意味を持たない。従って、あくまでも投機的な処理である。ただし、オンライン中にオフラインテストをして、故障プロセッサのシステムからの分離および故障プロセッサへのタスク割当て回避等の自己修復をしているとみなすこともできる。

【0092】図16は、検索処理への適用例である。図16において、Sは現状態、i1は入力であり、3回の検索(判定)を繰り返した結果最終的な処理(または検索結果)を選択する例を示している。各枝(0または1)は、各ノードにおける判定結果であり、各葉は最終的な処理(または検索結果)である。現状態Sにおいて、例えばNOP状態のプロセッサが十分存在すれば、最終的な処理(または検索結果)を予め全て求めておけばTATの短縮となる。計算時間およびNOP状態のプロセッサが十分存在しない場合は、現在の状態から可能性のある処理(図16では、s1~s4)をランダムに(もしくは、図2に示すIWMに従い)選択する。適当な見込処理が選択されれば、検索時間の短縮につながる。

【0093】検索処理と同じように将来起こる可能性のある処理を前もって処理する適用例として、原子力発電等における危険予知または危険回避が考えられる。すなわち、現状態において、例えばNOP状態のプロセッサが存在すれば、最終的な処理(または先の結果)を予め求めることにより、危険または故障につながる選択を予知し回避可能である。

【0094】以上、超並列マシンにおいて、NOP状態のプロセッサを活用して、任意に設定したプログラムを実行する技術に関して示した。変形例として、見込み処理用の専用プロセッサとして実現することも可能である。例えば、図2に示す要素プロセッサ120単体で使用することも可能である。

【0095】図19は、第3の実施例として、例えば温度、除湿機能、風量(強、中、弱)、風種(一定方向、左右、ランダム)をユーザ指定可能なエアコンをターゲットとした専用プロセッサ1900として実現した例を示している。見込み処理としては、エアコンの初期値として、同一環境条件(例えば、室温、湿度、時刻)における前回ユーザ設定値(温度、除湿機能、風量、風種)を用いて運転を開始することを想定している。

【0096】図19において、1910は見込み値メモリであり、ユーザが指定した設定値(温度、除湿機能、風量、風種)を格納する書き換え可能な記憶装置である。

【0097】初期値は、予め標準的なデータを記憶させ

ておく。あるいは、エアコン購入時に、ユーザがセットできる仕様とする。1920は、室温、湿度等の環境条件および時刻(曜日)を基に見込み値メモリ1910のポインタを生成する論理である。1930はタイマ、1940はシステム外部から入力した環境情報を解析し、室温、湿度等の環境条件を出力する論理、1950はシステム外部から環境情報を入力するためのセンサである。

【0098】次に、図19を用いてLSI1900の動作を簡単に説明する。エアコンのスイッチがオン状態となると同時に、LSI1900は室温、湿度等の環境情報を信号線1966によりシステム外部から入力し信号線1951を介して環境解析論理1940に伝達する。環境解析論理1940はセンサ1950を介して入力した環境情報を解析して室温、湿度等の環境データを適当なフォーマットに変換し、信号線1941を介して見込み値メモリポインタ決定論理1920に伝達する。見込み値メモリポインタ決定論理1920は、室温、湿度等の環境データおよび時刻(曜日)から見込み値メモリのポインタ1921を生成する。生成に当たっては、例えば室温、湿度および時刻を変数とする3次元テーブルを用いる。ここで時刻(曜日)を変数の一つとして考えるのは、時刻(曜日)によりスポーツ後、風呂あがり、食事中、睡眠中等のユーザ状態がある程度決まるのではないかと考えるからである。すなわち、外部環境条件およびユーザのコンディション(見込み状態)からエアコンの初期設定値を決定するためである。見込み値メモリ1910は、ポインタ1921で指定されるアドレスに格納されているデータを、それぞれ温度、湿度、風量、風種制御用の初期値信号1971~1974として出力する。初期値信号1971~1974は、それぞれそのまま温度、湿度、風量、風種制御用の信号1981~1984となる。見込んだ初期値が適切でなかった場合は、ユーザが温度、湿度、風量または風種の設定変更を指示するわけであるが、変更情報は信号線1960によりLSI1900に入力される。また、ユーザが温度、湿度、風量または風種の設定変更を指示をすると、信号線1965の値が(例えば0から1に)変化する。信号線1965の値が(例えば0から1に)変化する、見込み値メモリ1910から読み出された信号1971~1974に替わってユーザが指定した値1960(1961~1964)がそれぞれ温度、湿度、風量、風種制御用の信号1981~1984となる。また、ユーザが指定した設定値1960(1961~1964)がそれぞれ温度、湿度、風量、風種制御用のデータとしてポインタ1921で指定されるアドレスに格納される。(すなわち、見込み値メモリ1910の内容が更新される)エアコン等の家電製品は次第に高機能になり、ユーザ指定可能な要素も増加傾向にある。しかし、エアコンのスイッチをオンする度にこれらの設定をいちいち全て指定す

るのは面倒である。上記LSI1900を用いれば、外部環境条件およびユーザのコンディション（見込み状態）からエアコンの初期設定値（見込み値）を決定することが可能である。また、見込んだ初期値が完全には適切でなかった場合でも、一般的には一部の指定（例えば、風種）のみの変更でよい。

【0099】図20は、変形例として、初期チャンネルを見込みで設定するテレビをターゲットとした専用プロセッサ2000として実現した例を示している。見込み処理としては、テレビの初期チャンネル値として、同一環境条件（例えば、曜日、時刻）における前回ユーザ設定チャンネル値を用いて運転を開始することを想定している。

【0100】図20において、2010は見込み値メモリであり、ユーザが指定したチャンネル値を格納する書き換え可能な記憶装置である。初期値は、予め適当な値（視聴率の高い番組のチャンネル情報）を記憶させておく。あるいは、テレビ購入時に、ユーザがセットできる仕様とする。2020は、曜日および時刻を基に見込み値メモリ2010のポインタを生成する論理である。2030はタイマ、2035はタイマから入力した時刻に補正を行い、9:00PM、9:30PMのようにきりのよい時刻に変換する論理である。これは、例えばテレビのスイッチを8:56PMにオンした場合、ユーザが見たい番組は9:00PMから始まる番組と想像されるからである。2040はユーザが指定したチャンネル情報を解析し、時刻ごとにどのチャンネル（番組）を見ていたかを決定する論理、2050はユーザが指定したチャンネル情報をモニタリングする論理である。

【0101】次に、図20を用いてLSI2000の動作を簡単に説明する。テレビのスイッチがオン状態となると同時に、見込みチャンネル値メモリ2010は、ポインタ2021で指定されるアドレスに格納されているデータ（チャンネル値）をチャンネルの初期値として信号線2070に出力する。信号線2070の値はそのままチャンネル値信号（またはチャンネル制御信号）2080となる。ポインタ2021は、曜日および時刻の情報から見込み値メモリポインタ決定論理2020において生成される。生成に当っては、例えば曜日および時刻を変数とする2次元テーブルを用いる。曜日の情報はタイマ2030から信号線2032を介して入力する。時刻は補正論理2035から信号線2036を介して入力する。補正回路2035はタイマ2030から信号線2031を介して入力した時刻に補正を施し、きりのよい値（通常テレビ番組が始まる時刻）に変換し見込み値メモリポインタ決定論理2020に出力する。見込んだ初期チャンネル値が適切でなかった場合は、ユーザがチャンネルを設定し直す分けであるが、変更情報は信号線2060によりLSI2000に入力される。また、ユーザがチャンネルを設定し直すと同時に、信号線2065の値が（例えば0から1に）変化すると、見込みチャンネル値メモリ2010から読み出された信号2070に替わってユーザが指定した値（チャンネル情報）が信号線2080に伝達される。チャンネルモニタ2050はユーザが指定するチャンネル情報を観測し、信号線2051を介してチャンネル決定論理2040に出力する。チャンネル決定論理2040はチャンネルモニタ2050から入力する情報をもとに各時間帯ごとにユーザが見ていた番組のチャンネルを決定する。例えば、9:00PMから10:00PMの間で最も延べ時間の長いチャンネルをユーザが見ていた番組のチャンネルと判断する。決定されたチャンネル情報は、信号線2041を介して見込みチャンネル値メモリ2010に伝達され、ポインタ2021で指定されるアドレスに格納される。（すなわち、見込みチャンネル値メモリ2010の内容が更新される）現状、テレビのスイッチをオンすると前回スイッチをオフした時のチャンネルが初期値となっている。しかし、何曜日の何時には毎週どの番組を見るかを大体決めている人も多い。上記LSI2000を用いれば、ユーザが定期的に見る番組のチャンネル情報から、スイッチをオンした時のチャンネルの初期値（見込み値）を決定することが可能である。すなわち、見込みがうまく当れば、スイッチをオンした時点で自分の見たい番組にチャンネルがセットされていることになる。

【0102】以上、図19および図20により、同一環境条件における前回の値を基に見込みで初期値を決定し処理を開始する例を示した。このようなアイデアは他の家電品にも適用可能である。例えば、ビデオにおいて、先週の予約情報を参考に今週の予約を（見込みで）自動的に設定し、ユーザの意図と異なる部分だけ修正する使い方が考えられる。

#### 【0103】

【発明の効果】以上のように、本発明によれば、超並列計算機において、

- (1) 稼働率の向上
- (2) 見込処理のスケジューリング（いつ、どのような見込処理を実行させるか）の自動化
- (3) 最低限必要な見込処理から、新たな見込処理の自己獲得
- (4) 上記(1)～(3)に加え、例えばTATの短縮等の、付加価値の提供が実現可能である。

【0104】従って、システムを構成する要素プロセッサ数を増加させた場合の、稼働率低下の問題の解決策の一つとなる。また、より複雑な（人間の右脳に近い）情報処理に対応可能となる。

#### 【図面の簡単な説明】

【図1】超並列計算機システム構成図を示す図である。

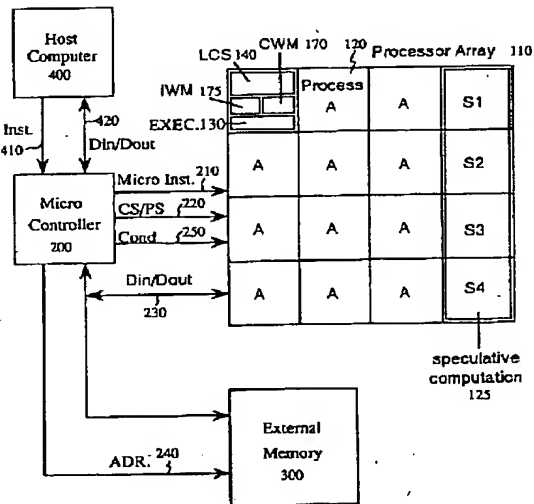
【図2】要素プロセッサの構成図を示す図である。

【図3】経験を通してしだいに適切な見込処理を学習するフローチャートを示す図である。

- 【図4】演算器群のブロック図の例を示す図である。  
 【図5】インストラクション決定論理の例を示す図である。  
 【図6】選択の成否判定を示す図である。  
 【図7】重み値（IWM）更新処理を示す図である。  
 【図8】空想的処理の実現手段の例を示す図である。  
 【図9】見込処理の実行および自律的な見込処理の獲得の例を示す図である。  
 【図10】自律的な見込処理の獲得フローを示す図である。  
 【図11】自律的な見込処理の獲得の例を示す図である。  
 【図12】空想的処理の遷移の例を示す図である。  
 【図13】見込処理と空想的処理の組み合わせの例を示す図である。  
 【図14】対話型処理への適用例を示す図である。  
 【図15】ニューロコンピュータの自己テストへの適用

【図1】

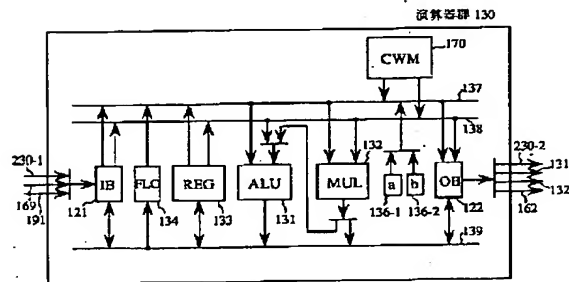
図1



- 例を示す図である。  
 【図16】検索処理への適用例を示す図である。  
 【図17】従来例（SIMD型超並列計算機）を示す図である。  
 【図18】従来例（命令プリフェッチの例）を示す図である。  
 【図19】エアコン初期値見込み設定用LSIブロック図を示す図である。  
 【図20】テレビ初期チャンネル見込み設定用LSIブロック図を示す図である。  
 【符号の説明】  
 110…プロセッサアレイ、200…マイクロコントローラ、210…マイクロ命令、140…局所記憶装置、175…インストラクションウエイトメモリ、165…命令決定論理、166…NOP判定論理、130…演算器群、190…選択した命令の成否判定論理。

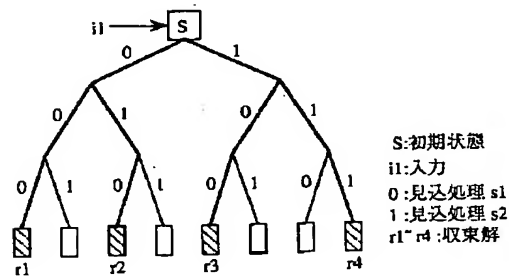
【図4】

図4



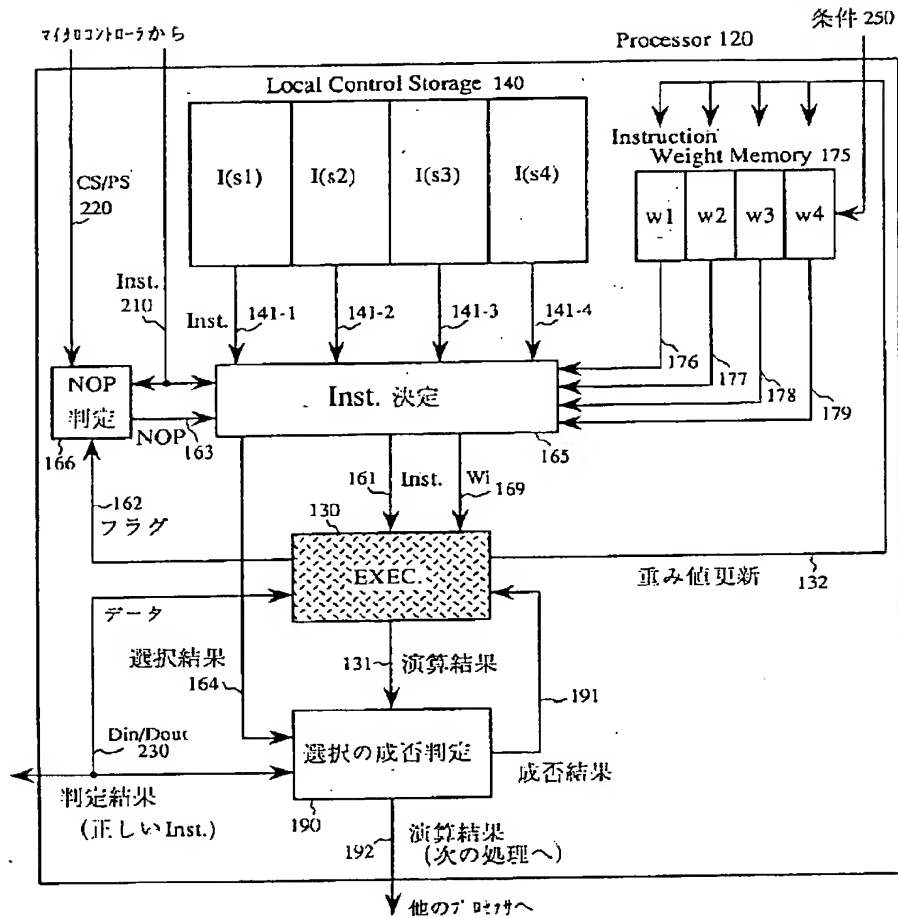
【図12】

図12



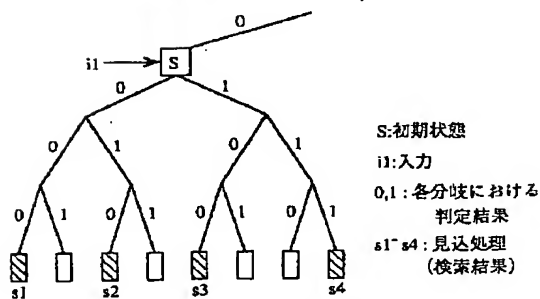
【図2】

図2



【図16】

図16



【图 13】

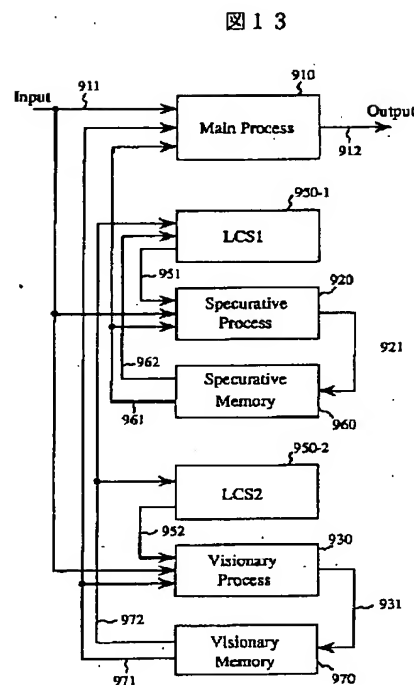
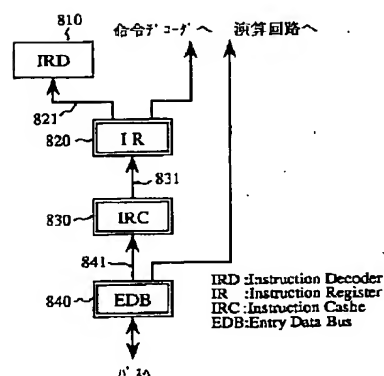
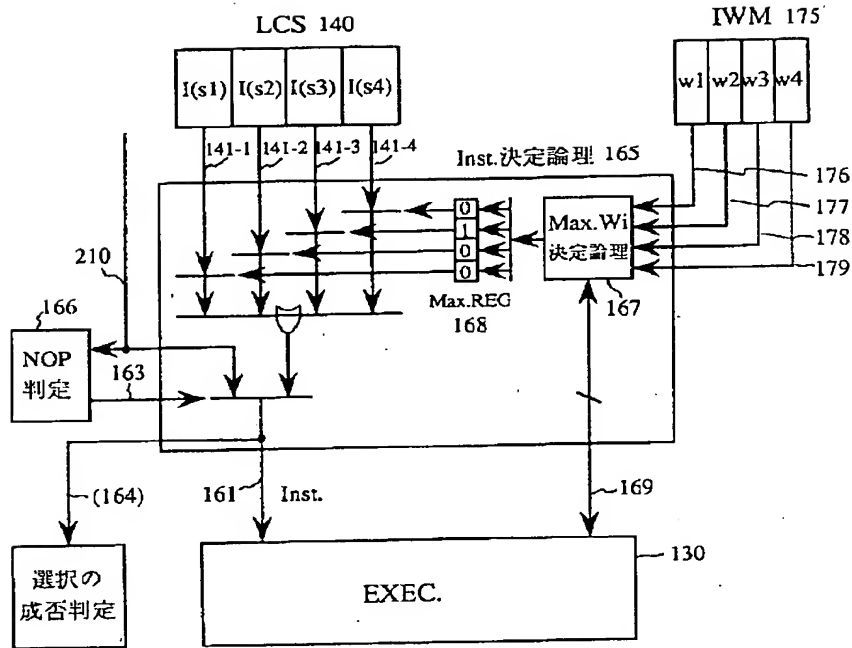


图 18



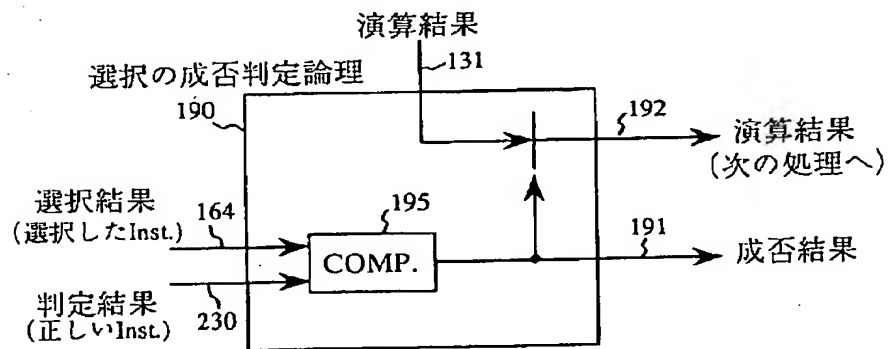
【図5】

図 5



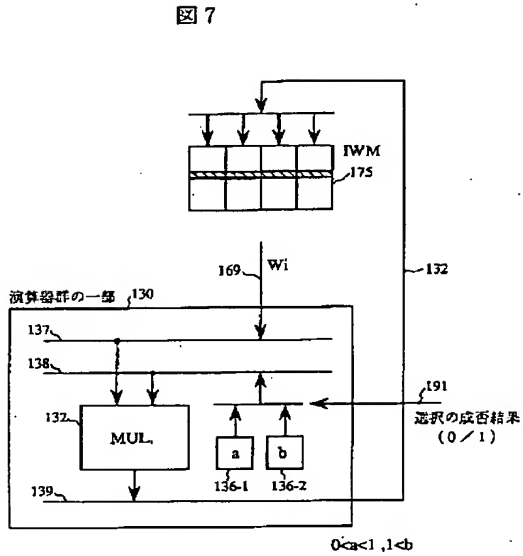
【図6】

図 6

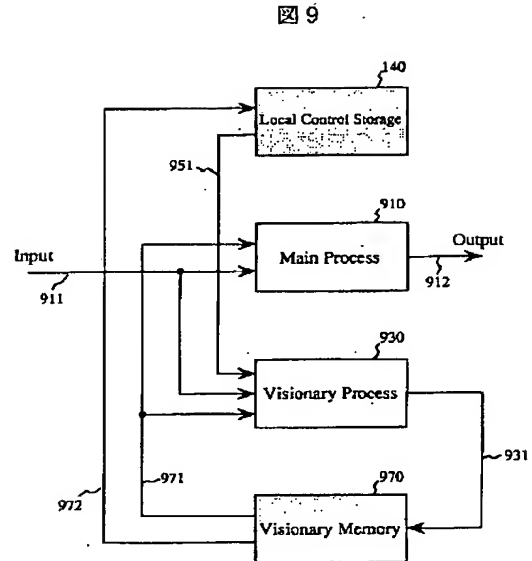




【図7】

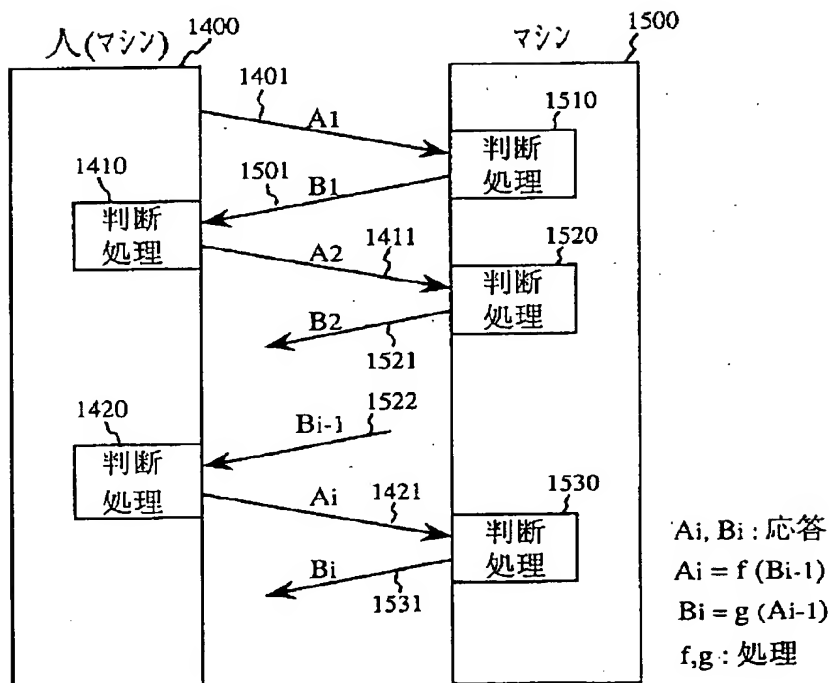


【図9】



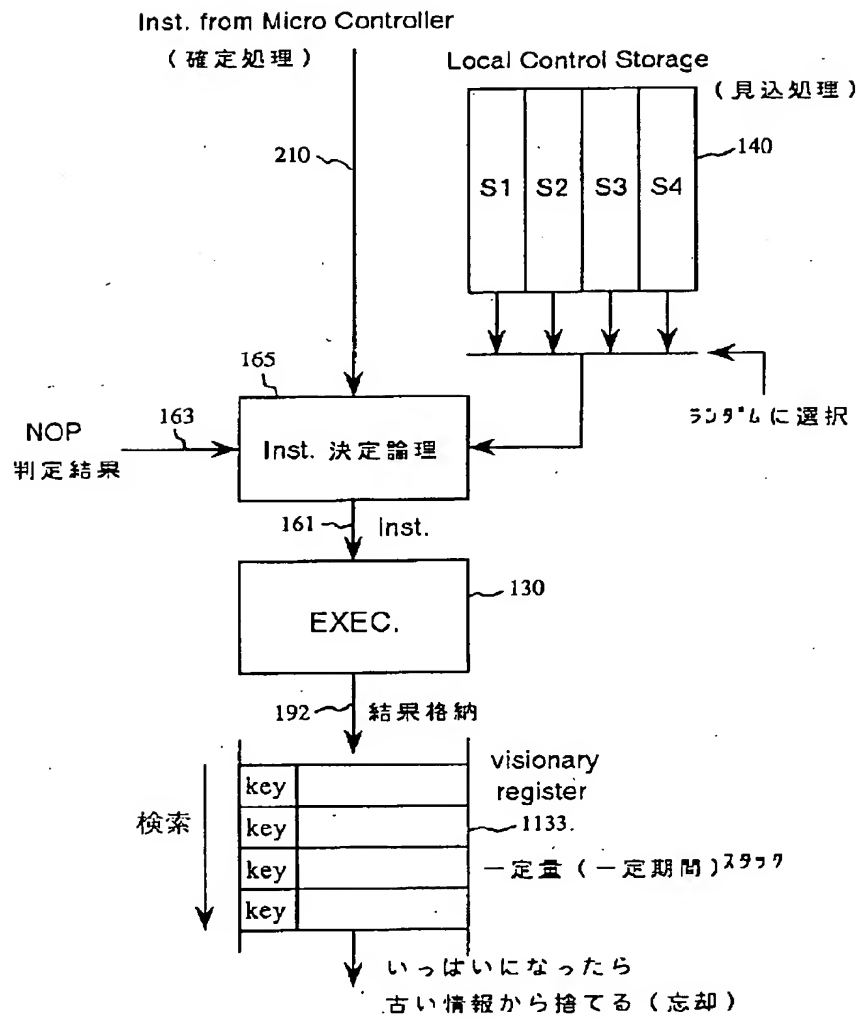
【図14】

図14



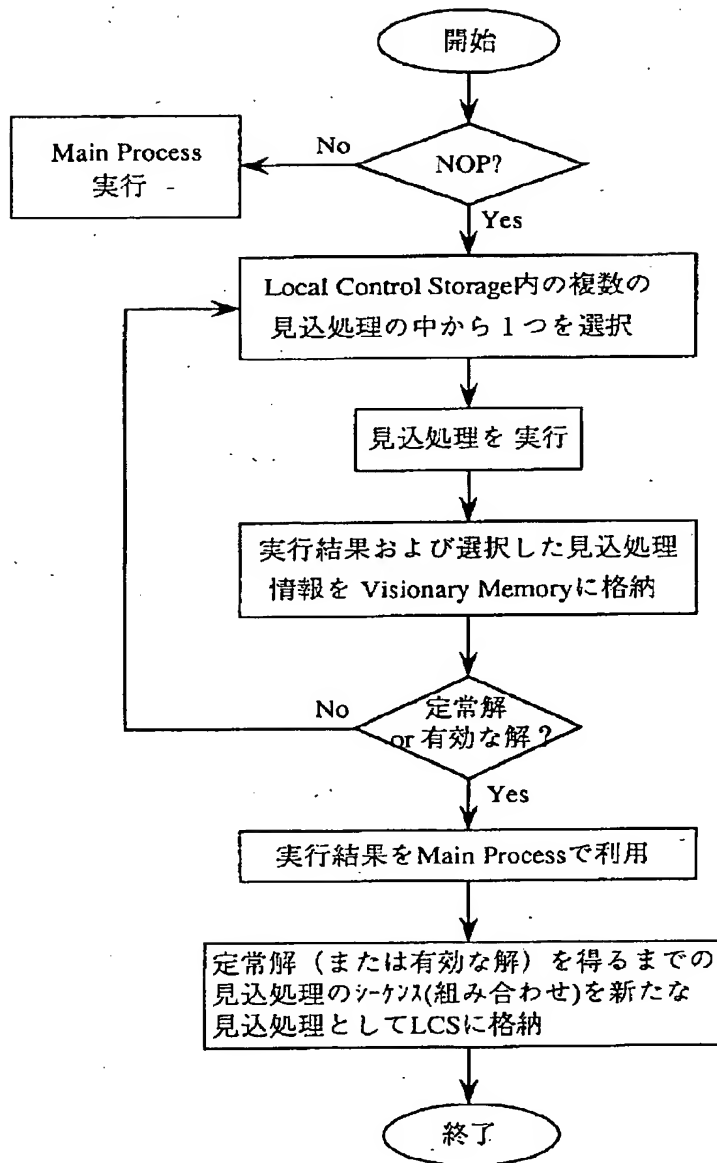
【図8】

図 8



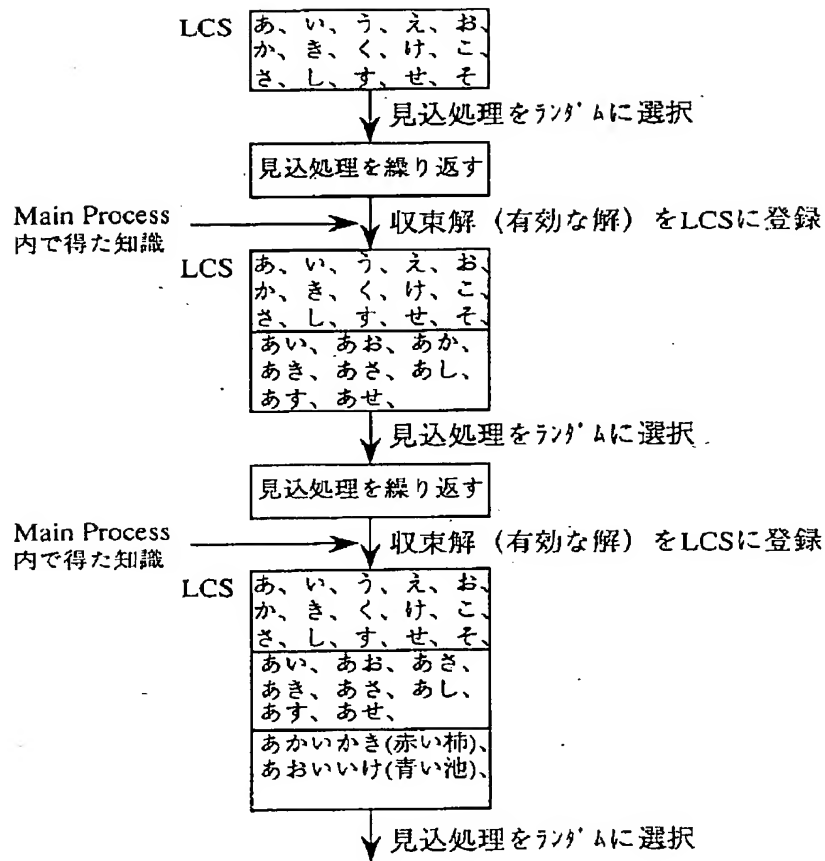
【図10】

図10.



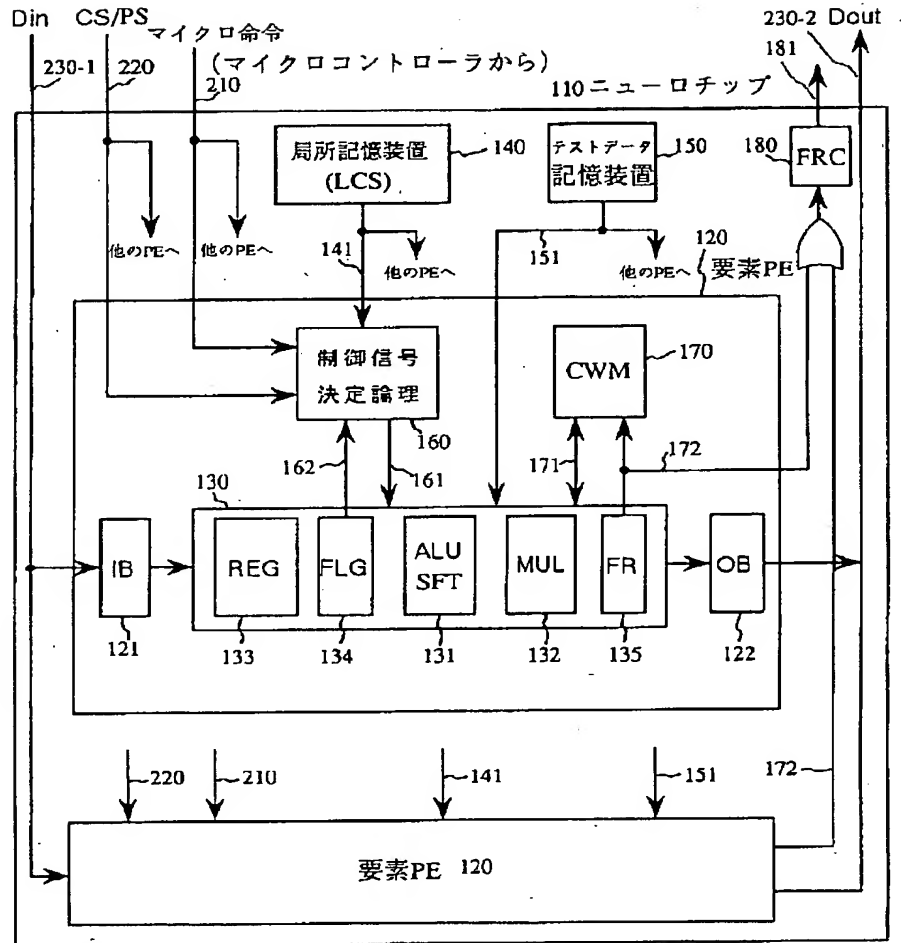
【図11】

図 1 1



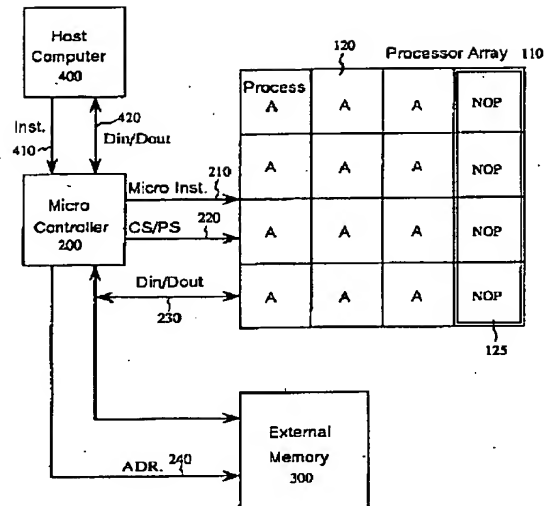
【図15】

図15



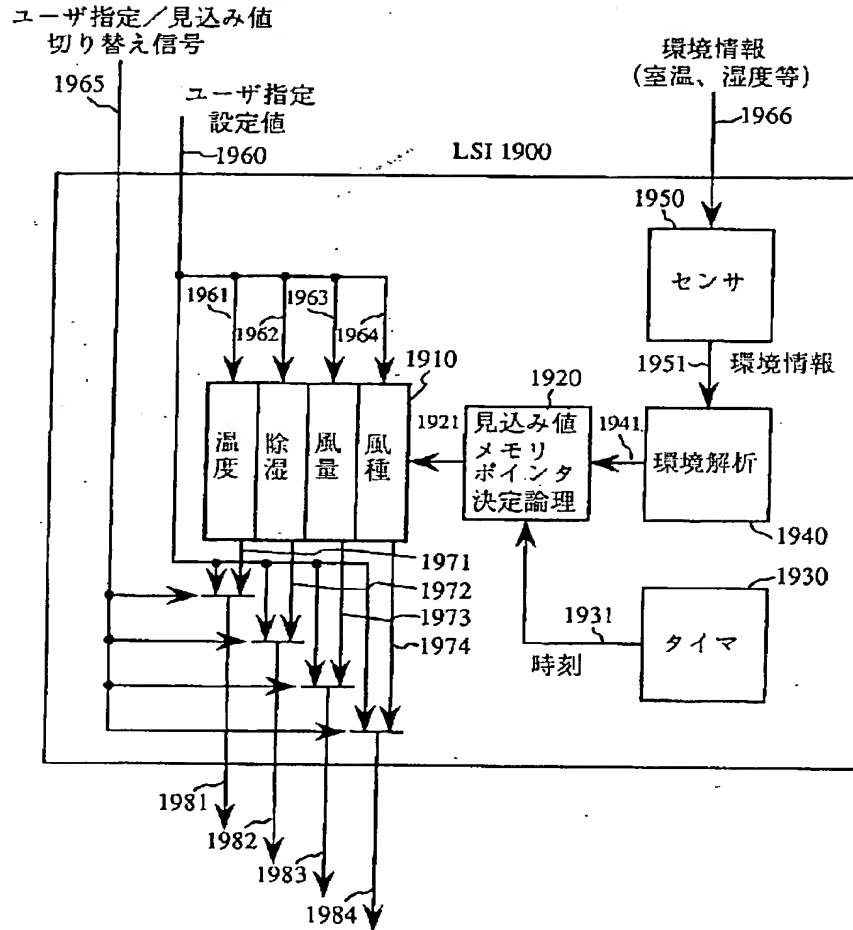
【図17】

図17



【図19】

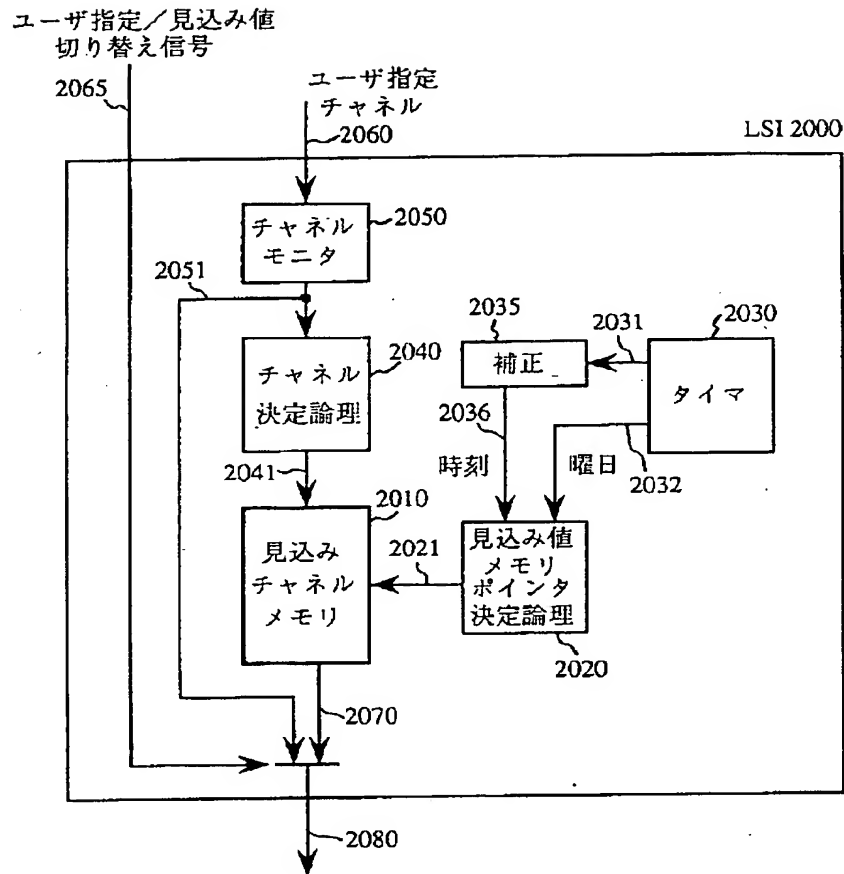
図19





【図20】

図20



フロントページの続き

(72)発明者 亀島 鉦二

茨城県土浦市神立町502番地 株式会社日立製作所機械研究所内

(72)発明者 圓佛 伊智朗

茨城県日立市久慈町4026番地 株式会社日立製作所日立研究所内

**\* NOTICES \***

Japan Patent Office is not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.

2.\*\*\*\* shows the word which can not be translated.

3.In the drawings, any words are not translated.

---

**Bibliography**

---

(19) [Country of Issue] Japan Patent Office (JP)

(12) [Official Gazette Type] Open patent official report (A)

(11) [Publication No.] JP,6-110853,A

(43) [Date of Publication] April 22, Heisei 6 (1994)

(54) [Title of the Invention] A parallel computer system and a processor

(51) [The 5th edition of International Patent Classification]

G06F 15/16 390 T 9190-5L

470 H 9190-5L

15/40 530 S 7218-5L

15/80 9190-5L

[Request for Examination] Un-asking.

[The number of claims] 27

[Number of Pages] 23

(21) [Filing Number] Japanese Patent Application No. 4-261108

(22) [Filing Date] September 30, Heisei 4 (1992)

(71) [Applicant]

[Identification Number] 000005108

[Name] Hitachi, Ltd.

[Address] 4-6, Kanda Surugadai, Chiyoda-ku, Tokyo

(72) [Inventor(s)]

[Name] Sato Yuji

[Address] 1-280, Higashi-Koigakubo, Kokubunji-shi, Tokyo Inside of Hitachi Central Lab

(72) [Inventor(s)]

[Name] Shikayama Masahiro

[Address] 4026, Kuji-cho, Hitachi-shi, Ibaraki-ken Inside of Hitachi Hitachi Lab

(72) [Inventor(s)]

[Name] Maeda Chapter

[Address] 1099, Ozenji, Asao-ku, Kawasaki-shi, Kanagawa-ken Inside of the Hitachi, Ltd. system development lab

(72) [Inventor(s)]

[Name] Kameshima Ore 2

[Address] 502, Kandatsumachi, Tsuchiura-shi, Ibaraki-ken Inside of Hitachi  
Mechanical Engineering Laboratory

(72) [Inventor(s)]

[Name] \*\*\*\* Ichiro

[Address] 4026, Kuji-cho, Hitachi-shi, Ibaraki-ken Inside of Hitachi Hitachi Lab

(74) [Attorney]

[Patent Attorney]

[Name] Brook Katsuo

---

[Translation done.]

**\* NOTICES \***

Japan Patent Office is not responsible for any  
damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not  
reflect the original precisely.

2.\*\*\*\* shows the word which can not be translated.

3.In the drawings, any words are not translated.

---

Summary

---

(57) [Abstract]

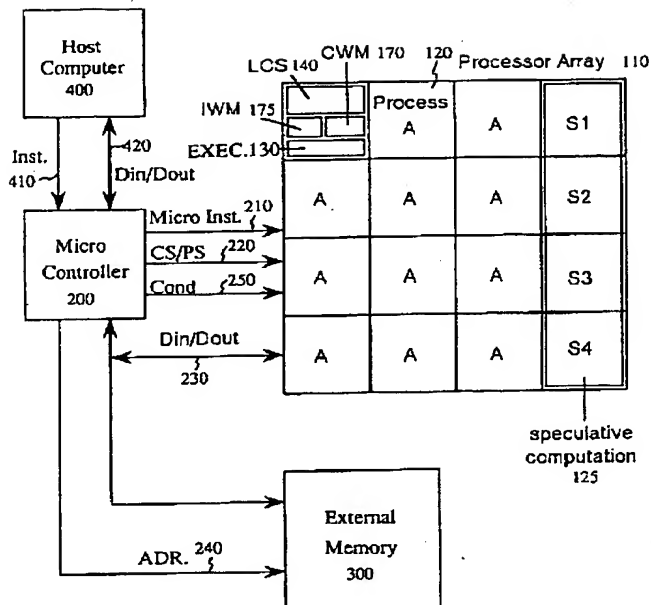
[Elements of the Invention] In the parallel computer system which has the  
microcontroller 200 which controls the processor array 110 which consists of two or  
more element processors 120, and the processor array 110, the result performed  
now has at least one for the program for chance [ that it does not know whether it  
will be helpful in the future ] processing, and performs [ any / one ] the program for  
prospective processing at least among two or more element processors 120.

[Effect] (1) Improvement in an operating ratio, and (2) Automation of the scheduling  
(what prospective processing is performed when?) of prospective processing, and (3)  
Self-acquisition of prospective processing to indispensable new prospective  
processing, and (4) The above (1) In addition to - (3), offer of added value of  
shortening of TAT etc. is realizable.

---

[Translation done.]

図 1




---

[Translation done.]

\* NOTICES \*

Japan Patent Office is not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.

2.\*\*\*\* shows the word which can not be translated.

3.In the drawings, any words are not translated.

---

CLAIMS

[Claim(s)]

[Claim 1] The parallel computer system by which the result performed now is characterized by having at least one for the program for chance [ that it does not know whether it will be helpful in the future ] processing, and performing the program for the above-mentioned prospective processing by any one at least among two or more above-mentioned processors in the parallel computer system which has the controller which controls the array which consists of two or more processors, and the array concerned.

[Claim 2] The parallel computer system according to claim 1 characterized by having a processing determination means to specify the program for one prospective processing, out of the program for two or more above-mentioned prospective processings.

[Claim 3] The parallel computer system according to claim 2 which has a means which consists of each above-mentioned processor or two or more processors to judge specific conditions for every group, changes to a main program according to the specific condition, and is smoothly characterized for making into a processing object the program for chance [ that above-mentioned specification was carried out ] processing by things with the feature.

[Claim 4] The above-mentioned specific conditions are a parallel computer system according to claim 3 characterized by being an NOP (No Operation) instruction.

[Claim 5] The parallel computer system according to claim 2 or 3 characterized by having an operation means for performing the program for chance [ that above-mentioned specification was carried out ] processing.

[Claim 6] The parallel computer system according to claim 5 characterized by having an evaluation means to evaluate the result by which the operation was carried out [ above-mentioned ] in accordance with specific criteria.

[Claim 7] A processing determination means to specify the program for one prospective processing is a parallel computer system according to claim 2 characterized by specifying the program for prospective processing using a table of random numbers out of the program for two or more above-mentioned prospective processings.

[Claim 8] The above-mentioned table of random numbers is a parallel computer system according to claim 7 characterized by realizing using a pseudo-random binary sequence counter.

[Claim 9] A processing determination means to specify the program for one prospective processing is a parallel computer system according to claim 2 characterized by specifying the program for the prospective processing which has the reliability information corresponding to each prospective processing, and should be performed based on reliability information out of the program for two or more above-mentioned prospective processings.

[Claim 10] The parallel computer system according to claim 9 characterized by having a means to update the reliability information corresponding to each above-mentioned prospective processing in online (operation) according to the above-mentioned evaluation means.

[Claim 11] The parallel computer system according to claim 5 characterized by having a means to memorize the result by which the operation was carried out [ above-mentioned ], and having a means to choose the result which searches the prospective processing result concerned and follows specific criteria.

[Claim 12] The information stored in a means to memorize the above-mentioned prospective processing result is a parallel computer system according to claim 11

characterized by adding and storing the labeling information for making reference easy etc. in the result of an operation.

[Claim 13] The operation means for performing the program by which specification was carried out [ above-mentioned ] is a parallel computer system according to claim 5 characterized by utilizing and performing the element processor of an NOP (No OPeration) state.

[Claim 14] The parallel computer system according to claim 1 characterized by applying to games and interactive-mode computers, such as the game of go, shogi, etc. to which the reply which processing advances and can be chosen by dialogic operation is restricted beforehand, supposing processing in which a reply of himself of a degree is beforehand prepared to the reply of some partners expected as the above-mentioned prospective processing.

[Claim 15] The parallel computer system according to claim 1 characterized by applying to a neuro computer and the improvement in reliability of a super parallel computer as the above-mentioned prospective processing supposing the online self-test of the NOP section.

[Claim 16] The parallel computer system according to claim 15 characterized by having the board which is a failure unit, a chip, or the function to separate a processor from a parallel computer system automatically (self-repair) when failure is detected as a result of the above-mentioned test.

[Claim 17] A means to separate the above-mentioned failure unit from a parallel computer system automatically (self-repair) is a parallel computer system according to claim 16 characterized by realizing by rewriting the value of the memory which memorizes the bond strength between processors.

[Claim 18] The parallel computer system according to claim 1 characterized by applying to TAT shortening of reference processing as the above-mentioned prospective processing supposing the final processing (or reference result) in reference processing.

[Claim 19] The parallel computer system according to claim 1 characterized by applying to dangerous precognition or risk aversion supposing the processing which may take place in the processing accompanied by risk as the above-mentioned prospective processing in the future.

[Claim 20] The parallel computer system according to claim 1 characterized by applying to the protein synthesis which suited the purpose as the above-mentioned prospective processing supposing the data expressing amino acid or amino acid.

[Claim 21] the parallel computer system according to claim 1 characterized by it being alike and applying to narrowing down the gene which suited the purpose as the above-mentioned prospective processing supposing the data expressing a gene or a gene

[Claim 22] The parallel computer system according to claim 1 characterized by having a means to acquire a new program autonomously based on the program for the above-mentioned prospective processing.

[Claim 23] A means acquire a new program autonomously based on the program for the above-mentioned prospective processing is a parallel computer system according to claim 22 characterized by realizing by storing in storage by considering as a new program suite-of-programs doubling for the above-mentioned prospective processing until it obtains the result with which the specific criteria are filled, as a result of repeating the program for the above-mentioned prospective processing two or more times.

[Claim 24] The processor characterized by making into a processing object at least one data with which effectiveness is not necessarily guaranteed in the processor aiming at predetermined control or predetermined initial value determination.

[Claim 25] the account of a top — the processor according to claim 24 characterized by having a means to memorize at least one data with which effectiveness is not necessarily guaranteed

[Claim 26] the account of a top — the processor according to claim 24 characterized by at least one data with which effectiveness is not necessarily guaranteed being a user specification value last time in the same environmental condition

[Claim 27] As the above-mentioned environmental condition, at least one in a day of the week is a room temperature, humidity, time, and a processor according to claim 26 characterized by containing.

---

[Translation done.]

**\* NOTICES \***

Japan Patent Office is not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.

2.\*\*\*\* shows the word which can not be translated.

3.In the drawings, any words are not translated.

---

**DETAILED DESCRIPTION**

---

[Detailed Description of the Invention]

[0001]

[Industrial Application] Generally this invention relates to parallel computer systems, such as a super parallel computer which consists of two or more processors. It is related with the composition of the operation of the super parallel computer for corresponding to information processing more complicated in a detail, and an



element processor.

[0002] this invention is applicable also to a neuro computer. Moreover, it can use also as a microprocessor simple substance.

[0003]

[Description of the Prior Art] A traditional computer system inputs a single instruction stream from program memory, and consists of processors of a simple substance which execute one instruction at a time. This is known as uniprocessor architecture (the von Neumann type architecture) one by one.

[0004] As the one technique of the improvement in the speed in uniprocessor architecture, the instruction prefetch (prefetch of instruction) function is indicated by for example, 68000 microcomputers (Maruzen, the March, 1983 issue, P.19-20) one by one. The easy register composition for realizing this function is shown in drawing 18 .

[0005] In drawing 18 , the instruction code under present execution is moved to IRD (Instruction Decoder)810 through a signal line 821, after being set to IR (Instruction Register)820 and decoding an instruction. If an instruction is moved to IRD810, from IRC (Instruction Cache)830, through a signal line 831, the instruction executed next will be moved to IR820, and will wait for the start of decoding. The first word of the instruction executed next is respectively set to IR820 and IRC830 in principle. The extension from the 2nd word of an instruction is set to EDB (Entry Data Bus)840. Thus, it is possible by having a special register for prefetch to take in advance and (read-out from memory) carry out the whole instruction executed next or a part, and to save instruction fetch time in execution.

[0006] however, the above -- the calculation speed and the throughput which a user desires are no longer acquired only by the improvement in a performance of the uniprocessor in a traditional computer system. The parallel computer (super parallel computer) system which realizes improvement in the speed by carrying out parallel execution of many element processors as this cure has attracted attention. As an example, the system configuration outline of the connection machine of sinking MASHINZU indicated by JP,60-84661,A is shown in drawing 17 .

[0007] In drawing 17 , it is the processor from which 110 constitutes a processor array and 120 constitutes a processor array. The microcontroller by which 200 controls a processor array, the external storage which stores the data which need 300 for an operation, and 400 are host computers which control a microcontroller 200.

[0008] A host computer 400 is a general purpose computer like the VAX computer of DEC (Digital Equipment Corp.) manufacture for example, marketed, and directs a start of operation and end of a microcontroller 200 with a control signal 410. Data are transmitted [ moreover, ] and received between external storage 300 or the processor array 110 by the signal line 420 through a microcontroller 200.

[0009] A microcontroller 200 controls the processor array 110 by the parallel bus 210. For example, one circuit in a bus 210 gives a RESET signal to an array 110,

three circuits take out a timing signal, and the remaining circuits are used for transfer of an instruction. The target chip and processor are performed by the signal line 220. An address signal 240 performs address information for reading/writing to external storage. A signal line 230 performs transmission and reception of the data between a microcontroller 200 and the processor array 110, and the notice of the status information from the processor array 110 to a microcontroller 200.

[0010] Buses 210 and 230 are connected in parallel with each element processor 120. Consequently, the signal from a microcontroller 200 is simultaneously given to each element processor 120 in an array 110. That is, each element processor 120 executes the same instruction at once. When you do not need all processors, the surplus processor 125 will be in an NOP (No OPeration) state.

[0011] The increase in the number of processors of an NOP state leads to the fall of cost performance. Therefore, the technology which will assign a load if NOP continues beyond the task (load) allocation technology that lessens the processor of an NOP state as much as possible, for example, a fixed period, is indicated by JP,3-191488,A etc.

[0012]

[Problem(s) to be Solved by the Invention] In order to correspond to a general more large-scale problem or more complicated information processing, it is necessary to make the number of the element processors which constitute a super parallel computer system increase.

[0013] When raising the degree of parallel and realizing improvement in the speed, it is necessary to make the number of processors increase. However, depending on an application program, all the processors in a system are not necessarily needed. Moreover, at the time of the subprogram (task) execution which constitutes a main program, a part of element processor which a main program needs is used. That is, the trouble that an operating ratio falls (the rate of the processor of an NOP state increases) exists with the increase in the number of processors which constitutes a system from above-mentioned conventional technology. This inclination is remarkable in especially the parallel computer of SIMD (Single Instruction stream Multi Data stream) type architecture by which all processors execute the same instruction at once.

[0014] Moreover, although it is thought that the read-ahead function (function to perform beforehand prospective processing (processing which may be helpful later supposing it carries out or, although it may not be helpful) in preparation for the function or future which predicts previous movement) which human being has is effective in order to correspond to more complicated information processing The above-mentioned conventional technology (for example, instruction prefetch function) is (not carrying out execution of processing) as the instruction which should next be executed in the already decided processing algorithm is beforehand read from memory, and it cannot respond to the read-ahead function which human being has enough. In order to realize a read-ahead function in the above-mentioned

conventional technology, people need to consider what read-ahead function is performed when, and it is necessary to write a main program. That is, an algorithm including the scheduling of a read-ahead function needs to be inquired, and a burden is large. every [ moreover, ] application program — the above-mentioned scheduling — it is necessary to re-evaluate — a programming man day — large — increasing — a bug — the trouble that the probability of occurrence also becomes large exists [0015] In order to solve the trouble of the above-mentioned conventional technology, it aims at (1) – (4) shown below in this invention.

[0016] (1) Improvement in the operating ratio of a super parallel computer (especially SIMD type architecture) (2) Automation of the scheduling (what read-ahead function is performed when?) of the above-mentioned read-ahead function (self-formation)

(3) Self-acquisition of prospective processing to indispensable new prospective processing (4) The above (1) It adds to – (3), for example, is offer [0017] of added value of shortening of TAT etc.

[Means for Solving the Problem] In order to attain the above-mentioned purpose, it has the solution means shown below.

[0018] (1) It has a means to memorize two or more programs (prospective processing) which are not understood whether an execution result is helpful (later) apart from a main program.

[0019] (2) It has a means to specify one program, out of two or more above-mentioned prospective processing objects. For example, it has the information (weight value) which shows each external condition (execution condition) of every to the probability of each prospective processing to the above-mentioned prospective processing, and one program is specified according to weight value information. Or one program is specified using a table of random numbers.

[0020] (3) It has an operation means for performing a chance [ that above-mentioned specification was carried out ] processing program.

[0021] (4) It has a means to evaluate the execution result of the above-mentioned prospective processing in accordance with specific criteria.

[0022] (5) It has a means which consists of each element processor or two or more element processors to judge an NOP (No OPeration) state for every group.

[0023] (6) According to the above-mentioned NOP state judging result, it changes to a main program and has a means to perform a chance [ that above-mentioned specification was carried out ] processing program.

[0024] (7) According to the evaluation result of the above-mentioned prospective processing, it has a means to update the above-mentioned weight value information in online (operation).

[0025] (8) it has the means which repeats the above-mentioned (the same — or it differs) prospective processing two or more times

[0026] (9) It has a means to memorize the career of the prospective processing repeated the account of a top.

[0027] (10) It has a means to memorize the combination of two or more prospective processings as new prospective processing.

[0028]

[Function] The operation shown below is obtained by the above-mentioned problem-solving means.

[0029] (1) After according to this invention memorizing two or more prospective processing programs in the system and specifying one out of two or more of these prospective processing programs apart from a main program, change to a main program in an NOP state in each element processor, and an operation of a chance [ that above-mentioned specification was carried out ] processing program is possible.

[0030] Therefore, an operating ratio improves by utilizing the processor of an NOP state for execution of prospective processing. If suitable prospective processing is performed, for example in reference processing etc., it will lead to shortening of TAT. Moreover, though invalid prospective processing is performed, since the processor of an NOP state is used from the first, it does not become degradation.

[0031] (2) According to this invention, evaluate the execution result of prospective processing in accordance with specific criteria, and update in online the information (weight value) which shows each external condition (execution condition) of every to the probability of each prospective processing according to this evaluation result.

[0032] Therefore, as it uses (letting experience pass), self-growth is more possible so that exact prospective processing may be chosen corresponding to an external condition (execution condition).

[0033] (3) As a result of repeating prospective processing two or more times, when an effective solution (convergence solution) is acquired according to this invention, combination of a series of prospective processings performed by the time it acquired the effective solution (convergence solution) can be considered as new prospective processing, and can be used.

[0034] Therefore, self-acquisition of new prospective processing (a certain initial state, flow of a series of prospective processings in which an effective solution (convergence solution) is given to a certain input data (put together)) is possible combining the prospective processing (hereditary element) which human being gave beforehand.

[0035]

[Example] Hereafter, drawing 20 explains the example of this invention from drawing 1.

[0036] Drawing 1 is the system configuration view of the super parallel computer of the SIMD (Single Instruction streamMulti Data stream) type architecture for one example of this invention being shown.

[0037] In drawing 1, it is the processor from which 110 constitutes a processor array and 120 constitutes a processor array. The microcontroller by which 200 controls a processor array, the external storage which stores the data which need

300 for an operation, and 400 are host computers which control a microcontroller 200.

[0038] A host computer 400 directs a start of operation and end of a microcontroller 200 with a control signal 410. Data are transmitted [ moreover, ] and received between external storage 300 or the processor array 110 by the signal line 420 through a microcontroller 200.

[0039] A microcontroller 200 controls the processor array 110 by microinstruction 210 (for example, specification of a calculation function). Chip select signal CS and the processor selection signal PS 220 perform target chip (a part of processor array or processor array) and processor (at drawing 1, since it assumes using a part of processor selection signal PS (for example, several bits high order) as chip select signal CS, CS and NS are expressed with the same signal 220.). Realizing as another signal is also possible. An address signal 240 performs address information for reading/writing to external storage. The Din/Dout signal 230 performs transmission and reception of the data between a microcontroller 200 and the processor array 110.

[0040] The difference from the conventional SIMD type super parallel computer shown in drawing 17 in drawing 1 is in the point of having the instruction weight memory (IWM) 175 which shows the reliability of two or more information (for example, program) stored in [ other than the computing-element group 130 ] the partial storage 140 and the partial storage 140 as an element which constitutes a processor 120, and the connection weight memory (CWM) 170 which shows the bond strength between processors. Cond250 is an external condition (or execution condition), and serves as a pointer of IWM175.

[0041] Hereafter, drawing 2 and drawing 3 show the feature of this invention.

[0042] Drawing 2 is an example of the block block diagram of the element processor 120 shown in drawing 1.

[0043] In drawing 2, 130, 140, and 175 are the computing-element group and partial storage which were shown in drawing 1, and IWM, respectively. In drawing 2, two or more programs (prospective processing) for which the partial storage 140 does not necessarily understand [ whether it is helpful (later) ] an execution result are memorized. The information (weight value) which shows the probability of each prospective processing is stored in each external condition (execution condition) of every at IWM175. The logic in which 166 generates NOP (No OPeration) judging logic, and 165 generates the control signal of the computing-element group 130 (on basis [ information / from IWM / NOP judging logic, partial storage, and ]), and 190 are logic which judges the effectiveness of the executed instruction.

[0044] The characteristic movement of the processor hereafter shown in drawing 2 based on the flow chart shown in drawing 3 is shown.

[0045] NOP judging logic performs the flag information 162 to the NOP judging inputted from CS/PS220, the microinstruction 210, and the computing-element group 130 which were inputted from the microcontroller 200. Moreover, the judgment

result 163 is outputted to the INST. decision logic 165.

[0046] The INST. decision logic 165 will output the microinstruction 210 inputted from the microcontroller 200 as instruction 161 for computing-element group control, if the judgment results 163 are not NOP directions. (That is, a main process is performed by the computing-element group 130.) When the judgment results 163 are NOP directions, one is chosen from prospective processings of the plurality (drawing 2 141- 1-141 four of -4) stored in the partial storage 140, and it outputs as instruction 161 for computing-element group control. Moreover, the information on the selected instruction is outputted to the success-or-failure judging logic 190 of selection. In drawing 2, it is determined which prospective processing is chosen from the weight value information (176-179) which shows the probability of each prospective processing read from IWM. The maximum of a weight value inputs and asks the computing-element group 130 for each weight value information through a signal line 169. The initial value of IWM is given to (it is random) with a table of random numbers. Or it has an intention and people determine. The pointer of IWM uses the external condition (execution condition) 250 inputted from the microcontroller 200.

[0047] The computing-element group 130 stores in storage, such as a register, as a result of performing an operation according to the instruction 161 inputted from the INST. decision logic 165 (and selected prospective processing information). Data required for an operation are beforehand inputted from a microcontroller 200 or external memory 300 through Din230, and are stored in the storage in the computing-element group 130.

[0048] In the success-or-failure judging logic 190 of selection, the effectiveness of the result of an operation 131 is judged from the information 230 on the instruction which should originally have been executed, and the selected instruction 164, for example. When effective, it transmits to other processors etc. through a signal line 192, and the result of an operation is used by the main process. Moreover, the success-or-failure judging result 191 of the selected instruction is outputted to the computing-element group 130.

[0049] The computing-element group 130 updates the weight value information corresponding to the selected instruction according to the success-or-failure judging result 191. For example, a weight value is made to increase when it is the right selection. A weight value is decreased when it is the mistaken selection. The weight value after updating is stored in IWM175 through a signal line 132.

[0050] Next, the detail of the component of an element processor shown in drawing 2 is shown using Table 1 and drawing 4 - drawing 7.

[0051] Table 1 is the table of truth value showing the detail of NOP judging logic.

[0052]

[Table 1]

表 1 . NOP 判定論理

入力			出力
CS/PS信号220	マイクロ命令信号210	フラグ信号162	NOP判定結果信号163
0	*	*	1
1	N O P	*	1
	N O P	0	0
	以外	1	1

\* : ドントケア

[0053] A CS/PS signal is un-choosing, when a signal value is '1' and it is selection and '0'. Flag signals are NOP directions at the time of '1'. An NOP judging result signal means carrying out the selection directions of the prospective processing for the instruction 210 from a microcontroller 200 at selection directions and the time of '1' at the time of '0'.

[0054] The example of block composition of the computing-element group 130 is shown in drawing 4 . In drawing 4 , 121 is an input buffer, 122 is an output buffer, and it has the function to store temporarily the data inputted from other element processors, respectively, and the data outputted to other element processors. For an arithmetic logic unit with shifter ability, and 132, as for a general-purpose register and 134, a multiplier and 133 are [ 131 / a flag register and 136-1 to 136-2 ] parameter registers. Moreover, 170 is memory in which rewriting for storing the weight value information which shows the bond strength between processors is possible. It reaches 136-1-136-2 and 170 can also be assigned to a part of general-purpose register 133. 137-139 are internal buses. 230-1,169,191 is Din/Dout and the weight value information which are shown in drawing, and as a result of [ of selection ] success or failure, respectively. 230-2,131. 132 and 162 are Din/Dout shown in drawing 2 , the result of an operation, the weight value information after updating, and flag information, respectively.

[0055] An example of the INST. decision logic 165 is shown in drawing 5 . The INST. decision logic 165 outputs the microinstruction 210 inputted from the microcontroller 200 as instruction 161 for computing-element group 130 control, when the NOP judging result signal 163 is '0'.

[0056] When the NOP judging result signal 163 is '1', one is chosen from the partial storage 140 to the prospective processing programs 141-1 to 141-4, and it outputs as instruction 161 for computing-element group 130 control. Moreover, the information 164 on the selected instruction is outputted to the success-or-failure judging logic 190 of selection. In drawing 5 , it has diverted as information 164 on the instruction which chose the instruction 161 for computing-element group 130 control as it was. In drawing 5 , it is determined which prospective processing is



chosen from the weight value information 176-179 which shows the probability of each prospective processing read from IWM175. The maximum of a weight value inputs and asks the computing-element group 130 for each weight value information through a signal line 169. For example, the difference of two weight values is searched for using ALU in the computing-element group shown in drawing 4, and a result judges size by positive or negative. A result will be registered into the maximum register 168 if the maximum in the weight value information 176-179 is obtained. '1' is set to the register corresponding to maximum in drawing 5.

[0057] An example of the success-or-failure judging logic 190 of selection is shown in drawing 6. The selected instruction 164 is compared with the information 230 on the instruction which 195 is a comparator and should originally have been executed in drawing 6, and when in agreement, value '1' is outputted to a signal line 191. '0' is outputted when inharmonious. The value of a signal line 191 is outputted to the computing-element group 130 as a success-or-failure result. Moreover, it is used as a signal which controls whether the result of an operation 131 is transmitted to other element processors through a signal line 192. It is also possible to use ALU instead of a comparator 195. The success-or-failure judging logic 190 (comparator 195) of selection is realizable even if it uses the computing-element group 130.

[0058] An example of a weight value (IWM) update process is shown in drawing 7. Drawing 7 shows the example which updates a weight value using the multiplier 132 and the parameter register 136-1 to 136-2 which are a part of computing-element group 130. The values  $a$  and  $b$  with which  $0 < a < 1$  and  $1 < b$  are filled are beforehand stored in a register 136-1 and 136-2, respectively. They are the value  $a$  stored in the parameter register 136-1 when the value of the success-or-failure result 191 inputted from the success-or-failure judging logic 190 of selection is '0', and a weight value corresponding to the selected prospective processing.  $W_i$  is inputted into a multiplier 132 through the internal buses 138 and 137, respectively, and the result of an operation is stored in IWM175 through a bus 139. It is the value  $b$  and weight value which are stored in the register 136-2 when the value of the success-or-failure result 191 is '1'.  $W_i$  is inputted into a multiplier 132 through the internal buses 138 and 137, respectively, and the result of an operation is stored in IWM175 through a bus 139.

[0059] Next, drawing 13 shows the 2nd example from drawing 8.

[0060] A main program is an example of the block block diagram for performing processing which drawing 8 almost became independent of.

[0061] It is the storage in which rewriting for the partial storage for 130 storing a computing-element group and 140 storing prospective processing and 165 storing an Inst. decision logic in drawing 8, and 1133 storing the result of an operation temporarily is possible.

[0062] When the NOP judging results 163 are NOP directions, the Inst. decision logic 165 changes to the microinstruction 210 inputted from the microcontroller 200, and performs prospective processing inputted from the partial storage 140. Since most

main programs assume the independent prospective processing, the difference from the processing shown by drawing 2 and drawing 3 is in the point which did not have the weight value information (IWM) 175 which shows the probability of each prospective processing, but has chosen at random the prospective processing which should be performed using a table of random numbers or a pseudo-random binary sequence counter.

[0063] The result of chance [ of having been processed by the computing-element group 130 ] processing is stored in visionary register1133, and is searched with suitable timing. If effective information by chance exists as a result of reference, it will be utilized by the main process. If visionary register1133 fills, it will delete from old information (oblivion). Or it deletes from the information that access frequency is low (oblivion). If visionary register1133 uses as a search key the prospective processing information chosen, for example and it carries out label attachment, it is effective in high-speed reference.

[0064] Drawing 9 shows the model which gains new prospective processing autonomously by repeating the above-mentioned prospective processing two or more times. The autonomous prospective processing acquisition which shows in drawing 9 hereafter based on the flow chart shown in drawing 10 is shown.

[0065] drawing 9 — setting — 910 — a main process — it is — processing (program which the algorithm has decided) usually finite based on an input 911 — carrying out — a result — 912 — outputting .

[0066] Selected prospective processing 951 is performed, choosing one at random out of two or more prospective processings in the partial storage 140 in the case of NOP, and using input data 911 if needed (930). An execution result and the selected prospective processing information 931 are stored in Visionary Memory970.

[0067] When the execution result stored in Visionary Memory970 is not a regular solution or an effective solution, by making this solution into the new input data 971, from the partial storage 140, one, new prospective processing is chosen at random and performed. Or prospective processing chosen previously is performed again. The above processing is repeated during a fixed period. If a regular solution or an effective solution is acquired by chance while repeating during a fixed period, the solution will be used by the main process. Moreover, it registers with the partial storage 140 by considering the sequence 972 of prospective processing until it acquires a regular solution (or effective solution) as new prospective processing.

[0068] The example of autonomous prospective processing acquisition is shown in drawing 11 . By drawing 11 , a character is given as initial value of prospective processing, and the flow which gains the text from the word and the word as new prospective processing is gradually shown from the character, making into a performance index the knowledge acquired by the main process.

[0069] Protein synthesis is also considered as same application. Amino acid can be considered as prospective processing, processing in which prospective processing is repeated (amino acid is combined) can be performed, and the combination of the

amino acid which forms the protein suitable for the purpose (evaluation function given from the outside) can be acquired. It is considered a kind of self-organizing. [0070] Drawing 9 can also be considered to be the narrowing-down model of prospective processing again. For example, a gene is considered as prospective processing. prospective processing is repeated (/a gene is rearranged -- combine) -- \*\* -- processing to say can be performed and it can leave only chance [ of suiting the purpose (for example, error criteria, such as a high nutrition strong against a pest, and safety) ] processing (a gene or a gene should put together) (it narrows down) It is considered a kind of Genetic Algorithm (genetic algorithm).

[0071] Next, an example of the information stored in Visionary Memory by drawing 12 and Table 2 is shown. Drawing 12 shows the example of changes of Visionary Process. By drawing 12, after considering only two, s1 and s2, as prospective processing, giving an input i1 to an initial state S and repeating s1 or s2 3 times, it is shown that the convergence solution of r1-r4 was acquired.

[0072] Table 2 shows an example of the information stored in Visionary Memory corresponding to above-mentioned drawing 12.

[0073]

[Table 2]

表 2. Visionary Memoryに格納する情報の一例

初期状態	入力	遷移	結果
S	i 1	0 0 0	r 1
S	i 1	0 1 0	r 2
S	i 1	1 0 0	r 3
S	i 1	1 1 1	r 4
:	:	:	:
:	:	:	:
:	:	:	:

遷移 0 : 見込処理 s 1 , 遷移 1 : 見込処理 s 2

[0074] For example, changes 000 (namely, prospective processings s1, s1, and s1) are performed, and the party eye of Table 2 shows that the convergence solution r1 was acquired as a result, after an input i1 is given to an initial state S.

[0075] It means that Visionary Memory is searched with suitable timing and new prospective processing was gained by registering the changes (sequence of prospective processing) corresponding to the convergence solution into the partial storage 140.

[0076] Drawing 13 is the modification of the autonomous prospective processing acquisition model which shows in drawing 9. Supposing the case where the

prospective processing which considered a main process and causal relation a certain grade shown by drawing 2 by drawing 13 , and chance [ of hardly considering a relation with the main process shown by drawing 8 ] processing live together, the former is called prospective processing and the latter is called imaginary processing. [0077] drawing 13 -- setting -- 910 -- a main process -- it is -- processing (program which the algorithm has decided) usually finite based on an input 911 -- carrying out -- a result -- 912 -- outputting .

[0078] Selected prospective processing 951 is performed, choosing one at random out of two or more prospective processings in the partial storage 140-1 in the case of NOP, and using input data 911 if needed (920). An execution result and the selected prospective processing information 921 are stored in Speculative Memory960.

[0079] When the execution result stored in Speculative Memory960 is not a regular solution or an effective solution, by making this solution into the new input data 961, from the partial storage 140-1, one, new prospective processing is chosen at random and performed. Or prospective processing chosen previously is performed again. The above processing is repeated during a fixed period. If a regular solution or an effective solution is acquired by chance while repeating during a fixed period, the solution will be used by the main process 910. Moreover, it registers with the partial storage 140-1 by considering the sequence 962 of prospective processing until it acquires a regular solution (or effective solution) as new prospective processing.

[0080] Or selected imaginary processing 952 is performed, choosing one at random out of two or more imaginary processings in the partial storage 140-2 in the case of NOP, and using input data 911 if needed (930). An execution result and the selected imaginary processing information 931 are stored in Visionary Memory970.

[0081] When the execution result stored in Visionary Memory970 is not a regular solution or an effective solution, by making this solution into the new input data 971, from the partial storage 140-2, one, new imaginary processing is chosen at random and performed. Or imaginary processing chosen previously is performed again. The above processing is repeated during a fixed period. If a regular solution or an effective solution is acquired by chance while repeating during a fixed period, the solution will be used by the main process 910. Moreover, it registers with the partial storage 140-1 and 140-2 by considering the sequence 972 of prospective processing until it acquires a regular solution (or effective solution) as new prospective processing.

[0082] As mentioned above, the example shown in drawing 13 from drawing 1 showed the example which utilizes the processor of an NOP state for execution of prospective processing (imaginary processing). This is because it does not become degradation since the processor of an NOP state is used from the first, though invalid prospective processing (imaginary processing) is performed.

[0083] It is also possible to recognize a certain external condition (execution condition), and to perform prospective processing (imaginary processing) instead of

NOP. It is also possible to assign the processor of exclusive use to prospective processing (imaginary processing).

[0084] As mentioned above, although the SIMD type computer was mentioned as the example in the example shown in drawing 13 from drawing 1, it does not limit to especially a SIMD type computer. The same effect is expectable also by the MIMD type computer.

[0085] Moreover, it is not necessary to have especially the partial storage 140 that stores prospective processing per processor. It is also possible to have per processor group (for example, a chip and a board).

[0086] The above-mentioned this invention can consider various application by the applicable field (main process) and chance [ of assuming ] processing (imaginary processing). Hereafter, drawing 16 shows the example of application of this invention from drawing 14.

[0087] Drawing 14 is an example of application to the processing which advances by dialogic operation between people, a machine, or a machine pair machine. In drawing 14, 1400 is a man (or machine) and 1500 is a machine. 1401-1421 show the response to a machine 1500 from a man 1400, and 1501-1531 show the response to a man 1400 from a machine 1500. It is shown that 1410, 1420, and 1510-1530 make a judgment to an input, and they perform a certain processing. For example, in 1420, Function  $f_i$  is chosen corresponding to the judgment made to input  $B_i-1$ ,  $A_i1421$  is returned to a machine 1500. In 1530, Function  $g_i$  is chosen corresponding to the judgment made to the input  $A_i1421$ , and as a result of giving function conversion  $g_i$  to Input  $A_i$ ,  $B_i1531$  is returned to a man 1400.

[0088] Here, a reply (response) of a partner is limited to its responses then outputted to a degree, such as the world, for example, the game of go, where the function which can be chosen is restricted beforehand, and shogi. therefore, " -- it will become shortening of TAT if anticipation comes true well by repeating chance [ that will utilize and a reply of himself of a degree will be beforehand called preparation" for the processor of an NOP state to the reply of some partners expected ] processing (1 time or more) Or as a result of repeating such prospective processing, "(for oneself) in order to draw the best result", it becomes realizable [ the flow of the dialog which is not made to interrupt realization or thinking of a complicated function ] from a say viewpoint by determining the first itself's reply.

[0089] Drawing 15 is an example of application to the self-test of a neuro computer. The neural processing chip 110 shown in drawing 15 is constituted by the element PE 120 of plurality ( drawing 15 two pieces), the partial storage 140, the test-data storage 150, and the failure flag register FRC180 in a chip. As prospective processing, only supposing the self-test in online (operation), the microinstruction for a test is distributed to the partial storage 140 in each neural processing chip 110 independently [ the microinstruction 210 (from a microcontroller to an input) for normal operation ], and it is in the point which can selection execute the

microinstruction 210 for normal operation, and the instruction 141 for a test in drawing 15 . The partial storage 140 and the test-data storage 150 are constituted from rewritable memory (for example, RAM, EPROM, a register, etc.) by each, and can be rewritten by application or the task (processing). The element PE 120 is constituted in drawing 15 by the weight value storage CWM 170 which stores the weight value which is a bond strength between the computing-element group 130, an input buffer 121, an output buffer 122, the control signal decision logic 160, and a neuron. The control signal decision logic 160 is the logic which doubled the Inst. decision logic 165 shown in drawing 2 , and the NOP judging logic 166. The computing-element group 130 is constituted by the arithmetic logic unit 131 with shifter ability, the multiplier 132, the register group 133, the flag register 134, and the failure flag register FR135 in PE. In drawing 15 , since only the self-test in online (operation) is assumed as prospective processing, it does not have the memory IWM175 which stores the weight value information which shows the probability of each prospective processing.

[0090] An outline of operation is explained using drawing 15 . For example, when CS220 is 1 (normal operation), the control signal decision logic 160 outputs the microinstruction 210 inputted from the microcontroller 220 as instruction 161 for controlling the computing-element group 130. According to a control signal 161, the computing-element group 130 is inputted through an input buffer 121 from Din 230-1, and also it calculates the internal state of a neuron from the weight value data inputted through a signal line 171 from the input data and CWM170 from Element PE (neuron). Or a weight value is updated. The output of the data to other PE (neuron) is performed using Dout 230-2 through an output buffer 122. On the other hand, CS220 is at the time (or 1 and the input 162 from a flag register output the value of specification [ CS220 ], and the microinstruction for a test which inputted the control signal decision logic 160 through the signal line 141 from the partial storage 140 (for example, when it is 1) as instruction 161 for controlling the computing-element group 130.) of 0 (the chip concerned is un-choosing). According to a control signal 161, the computing-element group 130 inputs a test data through a signal line 151 from the test-data storage 150, and performs the functional test of each computing-element group. The expected value of the result of an operation is beforehand stored in a part of REG133 (address fixation). When failure is not detected as a result of a test, FR135 holds initial value (for example, 0). When failure is detected, FR135 updates a value (it sets to 1). Moreover, the value of the weight value in the weight value storage 170 is set as 0. That is, the failure generating PE (neuron) is separated from a system. The value of each failure flag register FR135 in PE takes OR logic, and is set to the failure flag register FRC180 in a chip. The value of FRC180 is transmitted to a microcontroller 200 as a fault detection signal 181. A microcontroller 200 will give troubleshooting directions to a host computer 400, if failure generating is detected with the fault detection signal 181. A host computer 400 performs the display of the notice of failure generating, and a failure generating

part to a user according to troubleshooting directions. Or failure unit (board, chip, or PE) information is given to a compiler, and a task (processing) is made not to be assigned to a failure unit.

[0091] Drawing 15 showed the example of application to the self-test of the NOP section in an on-line state above. The tested portion gets used with the back operating condition of sufficiently short time, and is effective as a place and an online test. If a time interval until it will be in an operating condition excels, it does not have a meaning as an online test. Therefore, it is strictly speculative processing. However, it can also be considered that an off-line test is carried out into online, and self-repairs, such as separation from the system of a failure processor and task allocation evasion to a failure processor, are carried out.

[0092] Drawing 16 is an example of application to reference processing. In drawing 16, S is in the present state, i1 is an input, and the example which chooses final processing (or reference result) as a result of repeating three reference (judgment) is shown. Each branch (0 or 1) is as a result of [ in each node ] a judgment, and each leaf is final processing (or reference result). In the present state S, if the processor of an NOP state exists enough and it will ask for final processing (or reference result) altogether beforehand, it will become shortening of TAT. when the processor of machine time and an NOP state does not exist enough, possible processing ( drawing 16 s1-s4) is chosen from the present state at random (or IWM shown in drawing 2 — following) If suitable prospective processing is chosen, it will lead to shortening of retrieval time.

[0093] The dangerous precognition or risk aversion in nuclear power generation etc. can be considered by making reference processing and processing which may take place similarly in the future into the example of application processed beforehand. That is, in the present state, if the processor of an NOP state exists, by asking for final processing (or result of the point) beforehand, the selection which leads to risk or failure can be foreknown, and it can avoid.

[0094] As mentioned above, in the super parallel machine, the processor of an NOP state was utilized and the technology of performing the program set up arbitrarily was shown. As a modification, it is also possible to realize as a special purpose processor for prospective processing. For example, it is also possible to use it with element processor 120 simple substance shown in drawing 2 .

[0095] Drawing 19 shows the example realized as a special purpose processor 1900 which used as the target temperature, a dehumidification function, air capacity (a little more than, inside, weakness), and the air-conditioner that can user specify \*\*\*\* (the fixed direction, right and left, random) as the 3rd example. It assumes starting operation as initial value of an air-conditioner as prospective processing using the user set point (temperature, a dehumidification function, air capacity, \*\*\*\*) last time in the same environmental condition (for example, a room temperature, humidity, time).

[0096] In drawing 19, 1910 is prospective value memory and is rewritable storage

which stores the set point (temperature, a dehumidification function, air capacity, \*\*\*\*) specified by the user.

[0097] Initial value makes standard data memorize beforehand. Or it considers as the specification which a user can set at the time of air-conditioner purchase. 1920 is logic which expects based on an environmental condition and time (day of the week), such as a room temperature and humidity, and generates the pointer of the value memory 1910. The logic which 1930 analyzes a timer and the environmental information which inputted 1940 from the system outside, and outputs environmental conditions, such as a room temperature and humidity, and 1950 are the sensors for inputting environmental information from the system exterior.

[0098] Next, operation of LSI1900 is briefly explained using drawing 19. While the switch of an air-conditioner will be in an ON state, LSI1900 inputs environmental information, such as a room temperature and humidity, from the system outside by the signal line 1966, and transmits it to the environment-analysis logic 1940 through a signal line 1951. The environment-analysis logic 1940 analyzes the environmental information inputted through the sensor 1950, changes environmental data, such as a room temperature and humidity, into a suitable format, expects them through a signal line 1941, and is transmitted to the value memory pointer decision logic 1920. The prospective value memory pointer decision logic 1920 is expected from environmental data and time (day of the week), such as a room temperature and humidity, and generates the pointer 1921 of value memory. In generation, the 3-dimensional table which makes a room temperature, humidity, and time a variable, for example is used. Time (day of the week) is considered as one of the variables here because it thinks that the user state of the sleep middle class will be decided to some extent by time (day of the week) during bath stage fright and a meal after a sport. That is, it is for determining the initial value of an air-conditioner from the condition (prospective state) of external-environment conditions and a user. The prospective value memory 1910 outputs the data stored in the address specified with a pointer 1921 as temperature, humidity, air capacity, and initial value signals 1971-1974 for \*\*\*\* control, respectively. The initial value signals 1971-1974 turn into the signals 1981-1984 for temperature, humidity, air capacity, and \*\*\*\* control as it is, respectively. Change information is inputted into LSI1900 by the signal line 1960 although a user directs temperature, humidity, air capacity, or setting change of \*\*\*\* when the expected initial value is not suitable. Moreover, if a user directs temperature, humidity, air capacity, or setting change of \*\*\*\*, the value of a signal line 1965 will change (for example, 0-1). If the value of a signal line 1965 changes (for example, 0-1), the value 1960 (1961-1964) which replaced with the signals 1971-1974 read from the prospective value memory 1910, and the user specified will serve as the signals 1981-1984 for temperature, humidity, air capacity, and \*\*\*\* control, respectively. Moreover, the set point 1960 (1961-1964) specified by the user is stored in temperature, humidity, air capacity, and the address specified with a pointer 1921 as data for \*\*\*\* control, respectively. (That is, the content of the



prospective value memory 1910 is updated) Home electronics, such as an air-conditioner, become highly efficient gradually, and the element in which user specification is possible is also increasing. However, it is troublesome to specify these setup altogether one by one, whenever it switches off an air-conditioner. If the above LSI 1900 is used, it is possible to determine the initial value (prospective value) of an air-conditioner from the condition (prospective state) of external-environment conditions and a user. Moreover, generally the expected initial value is good at change of only a part of specification (for example, \*\*\*\*), even case [ which was not completely suitable ].

[0099] Drawing 20 shows the example realized as a special purpose processor 2000 which used as the target as a modification television which sets up an initial channel by the hope. It assumes starting operation as an initial channel value of television as prospective processing using a user setting channel value last time in the same environmental condition (for example, a day of the week, time).

[0100] In drawing 20, 2010 is prospective value memory and is rewritable storage which stores the channel value specified by the user. Initial value makes the suitable value (channel information on a program with a high audience rating) memorize beforehand. Or it considers as the specification which a user can set at the time of television purchase. 2020 is logic which expects based on a day of the week and time, and generates the pointer of the value memory 2010. 2030 is logic which a timer and 2035 amend at the time inputted from the timer, and is changed at the good time of an as like 9:00PM and 9:30PM. It is because the program which a user wants to watch is imagined to be the program which begins from 9:00PM when this switches off television in 8:56PM. The logic which determines which channel (program) 2040 was looking at for every time by analyzing the channel information specified by the user, and 2050 are logic which carries out the monitoring of the channel information specified by the user.

[0101] Next, operation of LSI2000 is briefly explained using drawing 20. While the switch of television will be in an ON state, the prospective channel value memory 2010 outputs the data (channel value) stored in the address specified with a pointer 2021 to a signal line 2070 as initial value of a channel. The value of a signal line 2070 serves as the channel value signal (or channel-control signal) 2080 as it is. A pointer 2021 is expected from the information on a day of the week and time, and is generated in the value memory pointer decision logic 2020. In generation, the two-dimensional table which makes a variable a day of the week and time, for example is used. The information on a day of the week is inputted through a signal line 2032 from a timer 2030. Time is inputted through a signal line 2036 from the amendment logic 2035. The amendment circuit 2035 amends at the time inputted through the signal line 2031 from the timer 2030, is changed into the good value (time when a TV program usually starts) of an as, and is outputted to the prospective value memory pointer decision logic 2020. Change information is inputted into LSI2000 by the signal line 2060 although it is the division to which a user resets a channel when the

expected initial channel value is not suitable. Moreover, the value of a signal line 2065 changes at the same time a user resets up a channel (for example, 0-1). Change (for example, 0-1) of the value of a signal line 2065 transmits the value (channel information) which replaced with the signal 2070 read from the prospective channel value memory 2010, and the user specified to a signal line 2080. The channel monitor 2050 observes the channel information specified by a user, and outputs it to the channel decision logic 2040 through a signal line 2051. The channel decision logic 2040 determines the channel of the program which the user was watching for every time zone based on the information inputted from the channel monitor 2050. For example, a channel with the longest manhour is judged to be the channel of the program which the user was watching between 9:00PM and 10:00PM. The determined channel information is expected through a signal line 2041, is transmitted to the channel value memory 2010, and is stored in the address specified with a pointer 2021. (That is, the content of the prospective channel value memory 2010 is updated) If the present condition and television are switched off, the channel when switching off last time serves as initial value. However, when of what day of the week also has many people who have decided which program to watch every week generally. If the above LSI 2000 is used, it is possible to determine the initial value (prospective value) of the channel when switching on from the channel information on the program which a user watches periodically. That is, the channel will be set to the program which he wants to watch, when it switches on, if the hope hit well.

[0102] As mentioned above, the example which determines initial value by the hope based on a value last time in the same environmental condition, and starts processing by drawing 19 and drawing 20 was shown. Such an idea is applicable to other household-electric-appliances articles. For example, in video, reservation of this week is automatically set as reference for reservation information last week (by hope), and the usage depending on which only a different portion from an intention of a user corrects can be considered.

[0103]

[Effect of the Invention] As mentioned above, according to this invention, it sets to a super parallel computer, and is (1). Improvement in an operating ratio (2) Automation of the scheduling (what prospective processing is performed when?) of prospective processing (3) Self-acquisition of prospective processing to indispensable new prospective processing (4) The above (1) In addition to - (3), offer of added value of shortening of TAT etc. is realizable.

[0104] Therefore, it is set to one of the solutions of the problem of an operating ratio fall at the time of making the number of element processors which constitutes a system increase. Moreover, correspondence in more complicated (close to human being's right brain) information processing is attained.

---

[Translation done.]

**\* NOTICES \***

Japan Patent Office is not responsible for any damages caused by the use of this translation.

1. This document has been translated by computer. So the translation may not reflect the original precisely.

2. \*\*\*\* shows the word which can not be translated.

3. In the drawings, any words are not translated.

---

**DESCRIPTION OF DRAWINGS**

---

[Brief Description of the Drawings]

[Drawing 1] It is drawing showing a super parallel computer system configuration view.

[Drawing 2] It is drawing showing the block diagram of an element processor.

[Drawing 3] It is drawing showing the flow chart which learns suitable prospective processing gradually through experience.

[Drawing 4] It is drawing showing the example of the block diagram of a computing-element group.

[Drawing 5] It is drawing showing the example of an instruction decision logic.

[Drawing 6] It is drawing showing the success-or-failure judging of selection.

[Drawing 7] It is drawing showing a weight value (IWM) update process.

[Drawing 8] It is drawing showing the example of the realization means of imaginary processing.

[Drawing 9] It is drawing showing execution of prospective processing, and the example of acquisition of autonomous prospective processing.

[Drawing 10] It is drawing showing the acquisition flow of autonomous prospective processing.

[Drawing 11] It is drawing showing the example of acquisition of autonomous prospective processing.

[Drawing 12] It is drawing showing the example of changes of imaginary processing.

[Drawing 13] It is drawing showing the example of the combination of prospective processing and imaginary processing.

[Drawing 14] It is drawing showing the example of application to interactive processing.

[Drawing 15] It is drawing showing the example of application to the self-test of a neuro computer.

[Drawing 16] It is drawing showing the example of application to reference processing.

[Drawing 17] It is drawing showing the conventional example (SIMD type super parallel computer).

[Drawing 18] It is drawing showing the conventional example (example of instruction prefetch).

[Drawing 19] It is drawing showing the LSI block diagram for an air-conditioner initial value prospective setup.

[Drawing 20] It is drawing showing the LSI block diagram for an initial channel [ of television ] prospective setup.

[Description of Notations]

110 [ -- Microinstruction, 140 / -- Partial storage, 175 / -- Instruction weight memory, 165 / -- An instruction decision logic, 166 / -- NOP judging logic, 130 / -- A computing-element group, 190 / -- Success-or-failure judging logic of the selected instruction. ] -- A processor array, 200 -- A microcontroller, 210

[Translation done.]

### \* NOTICES \*

Japan Patent Office is not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.

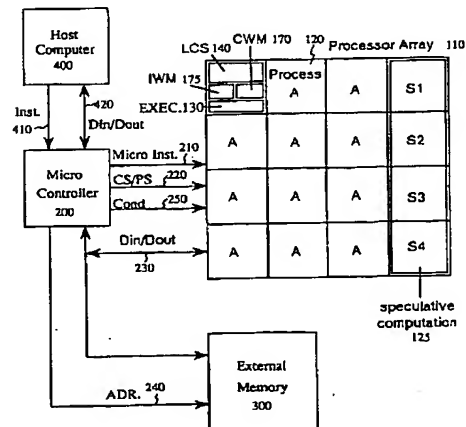
2.\*\*\*\* shows the word which can not be translated.

3.In the drawings, any words are not translated.

## DRAWINGS

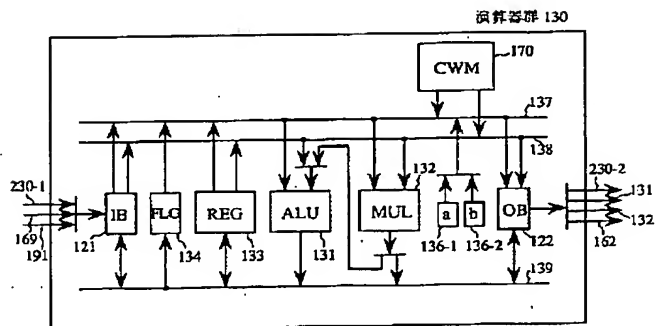
[Drawing 1]

図 1



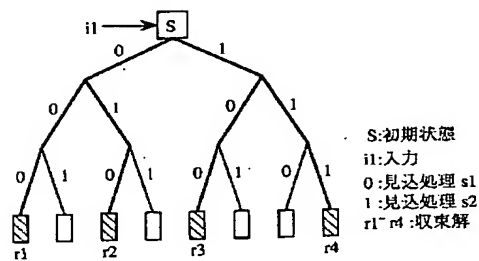
[Drawing 4]

図 4



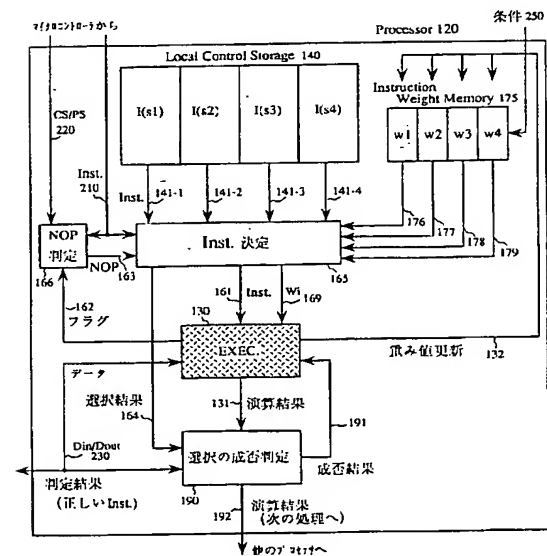
[Drawing 12]

図 1 2



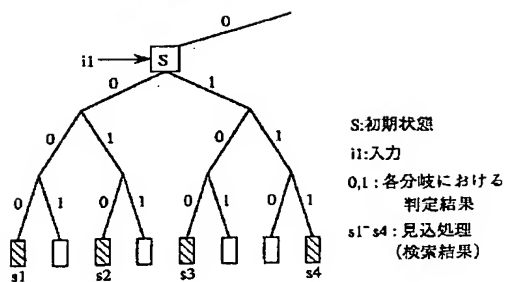
[Drawing 2]

図 2



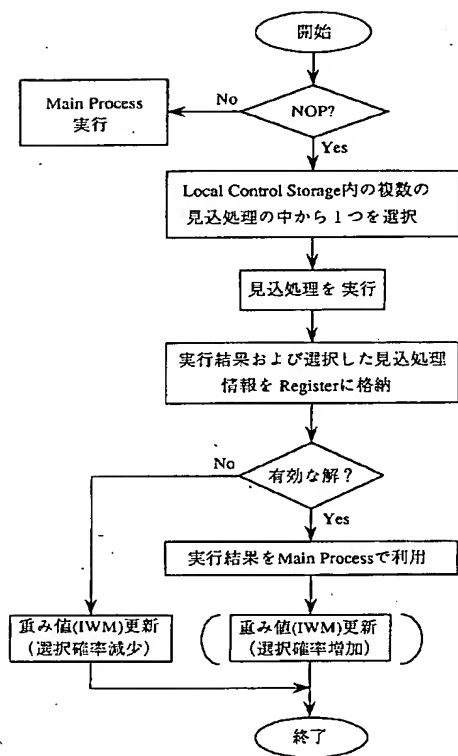
[Drawing 16]

図 16



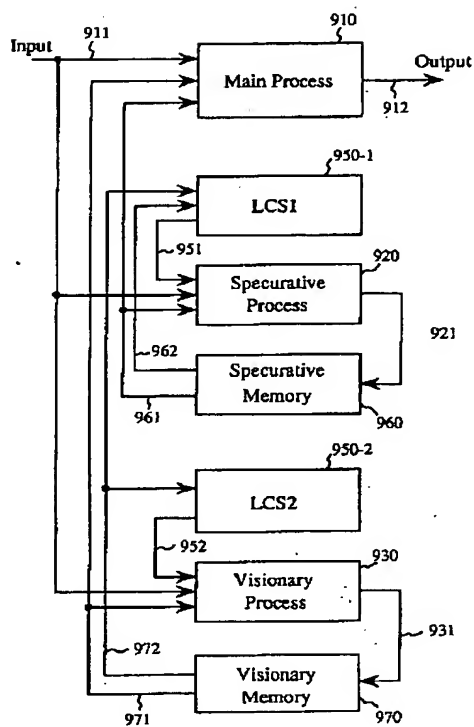
[Drawing 3]

図 3



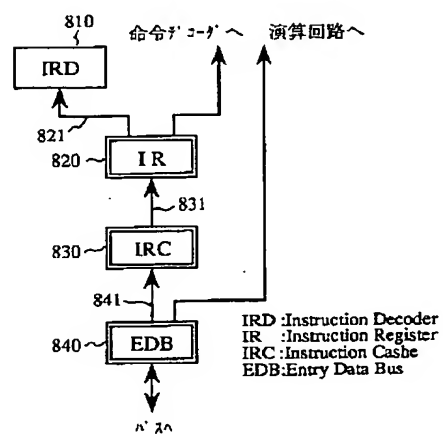
[Drawing 13]

図 13



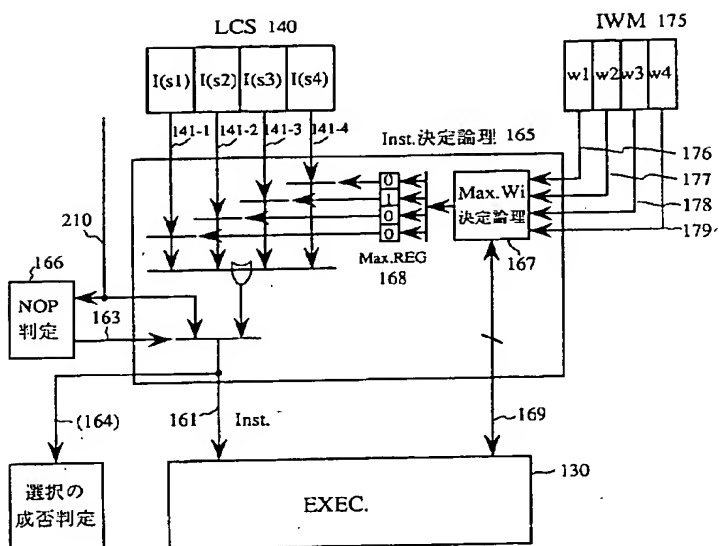
[Drawing 18]

図 18



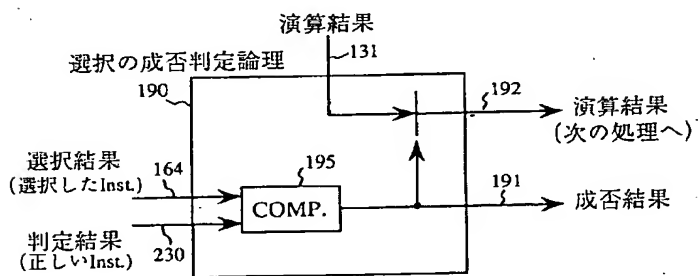
[Drawing 5]

图 5



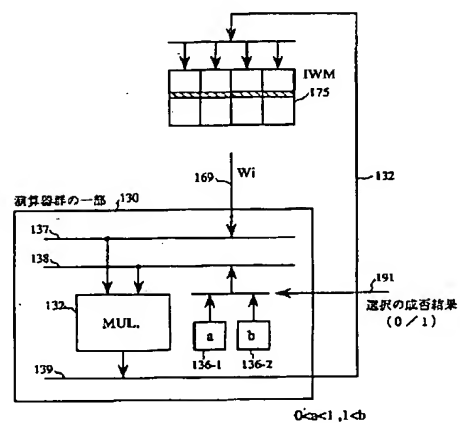
[Drawing 6]

图 6



[Drawing 7]

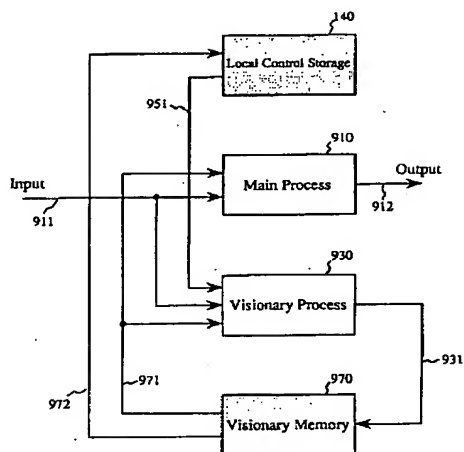
圖 7





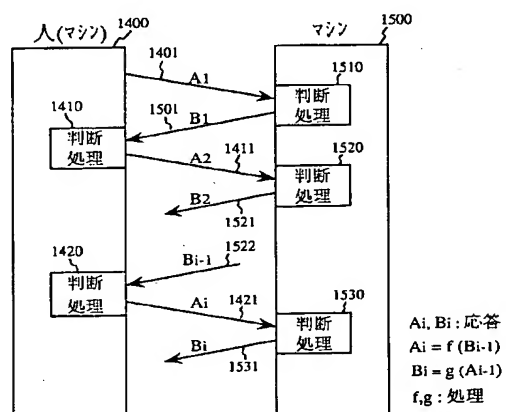
[Drawing 9]

図 9



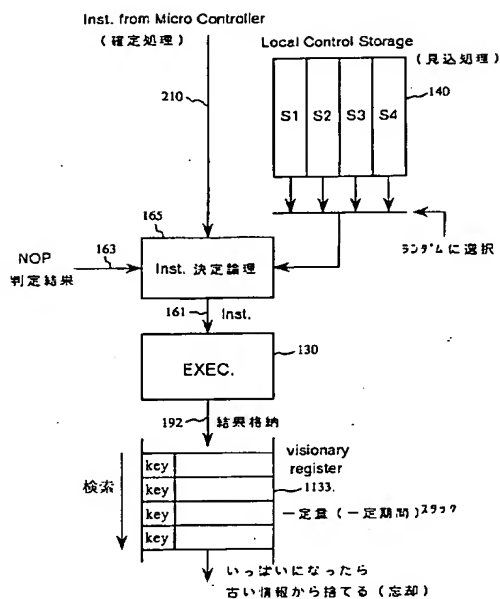
[Drawing 14]

図 14



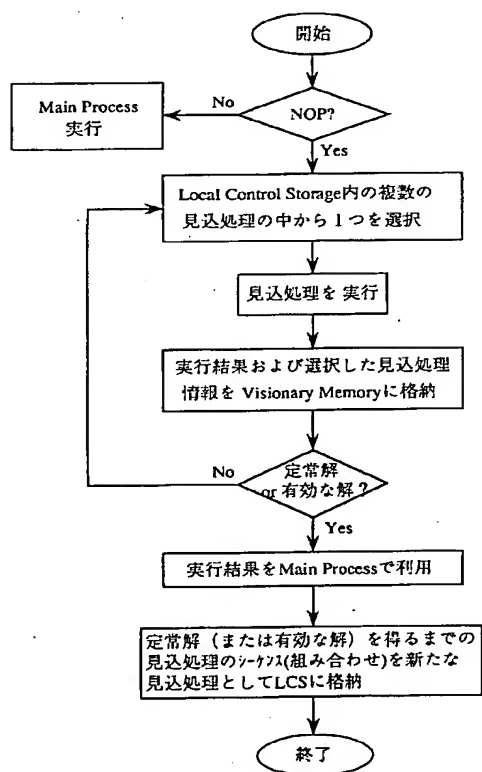
[Drawing 8]

図 8



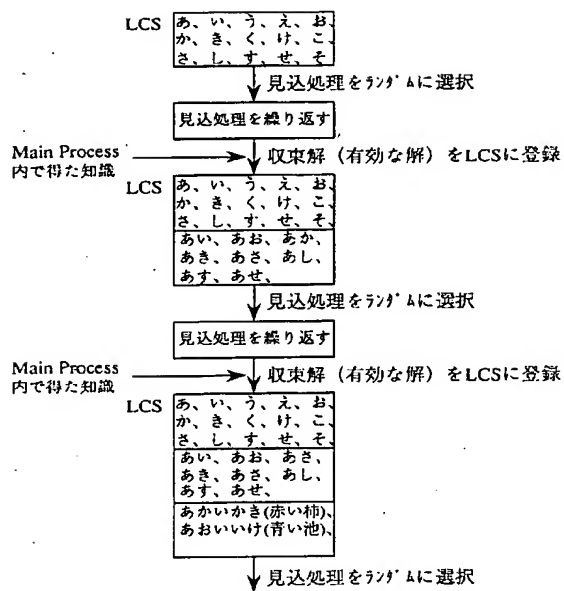
[Drawing 10]

図10.



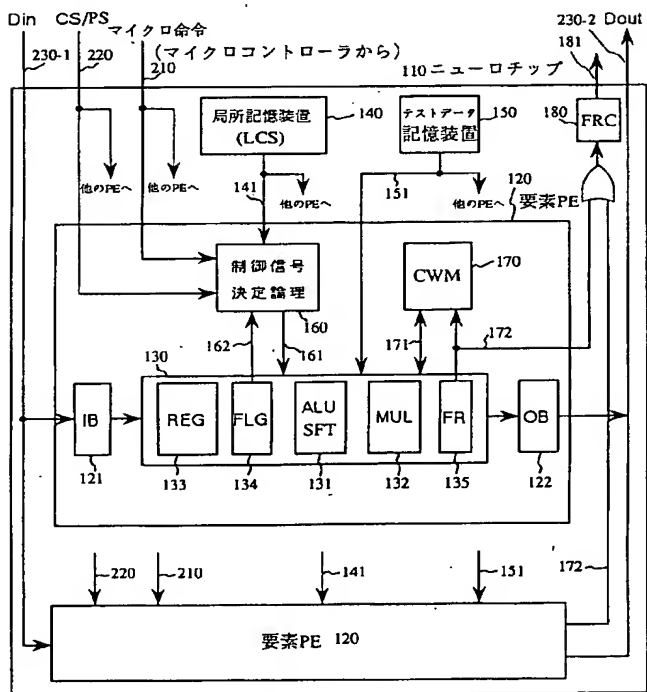
[Drawing 11]

图 1 1



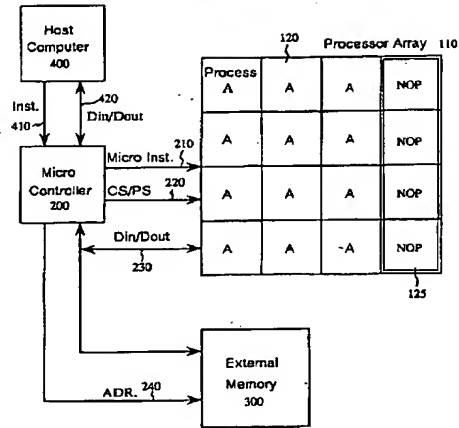
[Drawing 15]

图 15



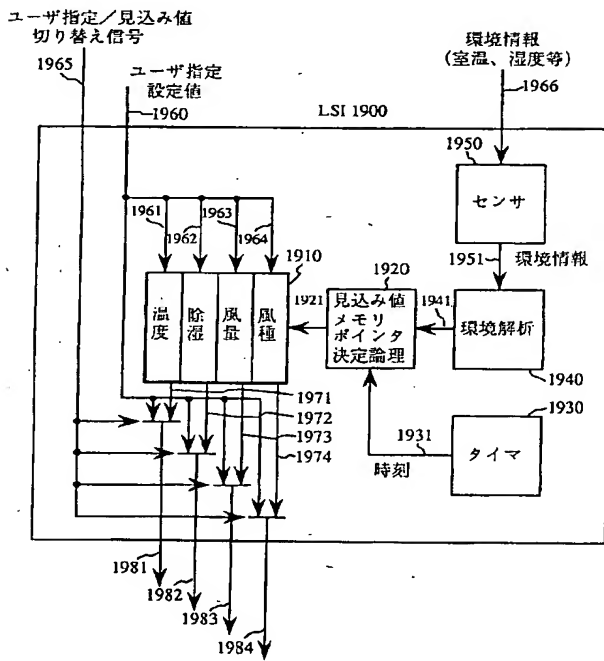
[Drawing 17]

図 17



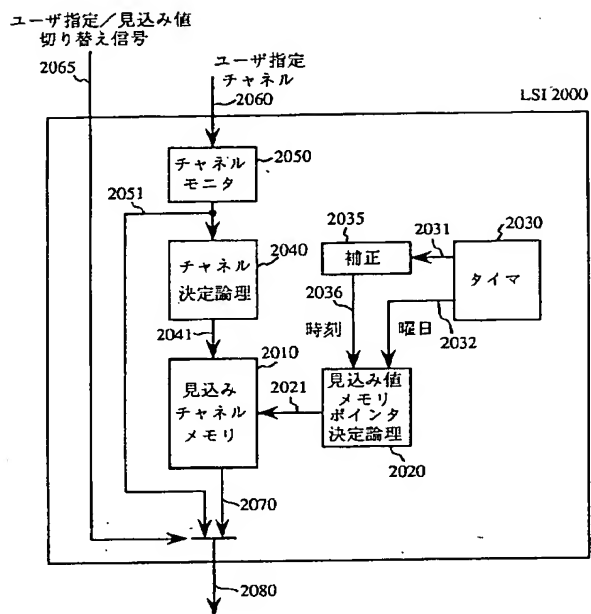
[Drawing 19]

図 19



[Drawing 20]

図 20



[Translation done.]